

# Software Technology Readiness Assessment

## Defense Acquisition Guidance with Space Examples

Dr. Peter Hantos  
The Aerospace Corporation

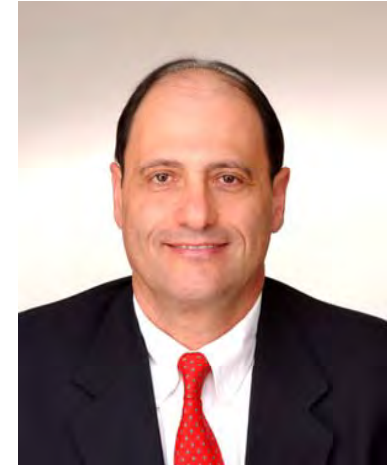
2010 Systems and Software Technology Conference,  
June 8-11, 2010

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE <b>APR 2010</b>		2. REPORT TYPE		3. DATES COVERED <b>00-00-2010 to 00-00-2010</b>	
4. TITLE AND SUBTITLE <b>Software Technology Readiness Assessment. Defense Acquisition Guidance with Space Examples</b>				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>The Aerospace Corporation,P.O. Box 92957-M1/112,Los Angeles,CA,90009-2957</b>				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release; distribution unlimited</b>					
13. SUPPLEMENTARY NOTES <b>Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License</b>					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT <b>Same as Report (SAR)</b>	18. NUMBER OF PAGES <b>82</b>	19a. NAME OF RESPONSIBLE PERSON
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>			

# About Your Instructor

## Dr. Peter Hantos

Senior Engineering Specialist  
The Aerospace Corporation  
P.O. Box 92957-M1/112  
Los Angeles, CA 90009-2957  
Phone: (310) 336-1802  
EMail: [peter.hantos@aero.org](mailto:peter.hantos@aero.org)



### Degrees Received:

Doctorate in Automation, Technical University, Budapest, Hungary  
M.S. in Electrical Engineering, Technical University, Budapest, Hungary

### Industry Experience:

Over 35 years of combined experience as professor, researcher, software engineer and manager, with accomplishments in software engineering, manufacturing automation, office automation and signal processing. As a senior software engineer at Xerox he was member of the original development team that was chartered to bring into the product development domain the world-known inventions from the Xerox Palo Alto Research Center (PARC), primarily the use of the desktop, icons, mouse-control and network paradigms. Successfully directed engineering and quality process development on all levels of the enterprise. As principal scientist of the Xerox Corporate Software Engineering Center developed and implemented two corporate-wide processes for software-intensive product development and the assessment of software technology readiness for productization. In the same capacity he also implemented a corporate-wide software risk assessment program.

### Government/Defense Program Support Experience:

NASA, FBI, CIA, Space Based Infra-Red System (SBIRS), Global Positioning System (GPS), Space Radar (SR), and National Systems Group Customers of The Aerospace Corporation

### Teaching Experience:

GSAW 2005, 2006, 2007, 2008, 2009, 2010 Tutorials; Euro-SEPG 2005, INCOSE 2005, NDIA 2008 and 2009 Tutorials; "Space System Software Project Management", "Space System Software Acquisition Management", "Space System Software Product Development", "Space System Development, Integration, & Test", "Introduction to Program Office Data and Controls", and "Program Measurement Workshop" courses for The Aerospace Institute; Electrical Engineering and Computer Science courses at the Technical University, Budapest, Hungary, and at the University of California, Santa Barbara (UCSB). Dr. Hantos has also been authorized by the SEI to teach the "Introduction to CMMI®-DEV V1.2" SEI Course.

# Acknowledgements

- This work would not have been possible without the help of the following people of The Aerospace Corporation
  - *Suellen Eslinger*
  - *Dr. Robert Frueholz*
  - *Dr. Leslie Holloway*
- A special thanks goes to the members of the Air Force Smart Operations for the 21<sup>st</sup> Century/Develop and Sustain Warfighting Systems/Technology Development Team
  - *Dr. Thomas Christian (AFMC/ASC)*
  - *Suzanne Garcia (SEI)*
  - *William Nolte (AFMC/AFRL)*
  - *Dr. Paul Phister (AFMC/AFRL)*
  - *Dr. Kyle Yang (MIT/Lincoln Lab)*

# The Technology Readiness Challenge

When the nation's first ballistic missile rose about 6 inches above the launch pad before toppling over and exploding, <Simon> Ramo reportedly turned to an Air Force general and said: "Well, Benny, now that we know the thing can fly, all we have to do is improve its range a bit."

~~~ Book Review, LA Times, July 5, 2009, B4 Business, by Peter Pae

---

*(Simon Ramo was the co-founder of and the "R" in TRW Corporation, now part of Northrop Grumman Corporation; TRW was acquired together with Litton, Westinghouse, Logicon, etc.)*

# Outline

- Motivation
- Technology Readiness Assessments – the 64,000-foot View
- Technology Readiness Assessments – the 10,000-foot View
- Software Technology Readiness Definitions
- Emerging Software Technologies to Watch
- Algorithms
- Environmental Considerations
- What Can We Learn from the Software Architecture?
- Selected, High-Level Viewpoints for Technology Readiness Assessment
- Critical Technology Element Identification Questions
- Software Technology Readiness Determination
- Case Studies
  - *Space Vehicle: Software Critical Technology Element Identification for a Space Vehicle*
  - *Ground System: Is Service-Oriented Architecture (SOA) a Critical Technology Element?*
- Challenging Topics
- Concluding Thoughts
- Acronyms
- References
- Backup Slides

# Motivation

- Why is Technology Readiness Assessment important?
  - *“The inability to define and thus measure technology readiness facilitates decisions to incorporate immature technology in system design at Milestone B which consequently leads to technical problems during System Design and Development.” [DAPA 2006]*
- Why should it be important for You?
  - *For one thing, it is the Law (more on this later.) Nevertheless ...*
  - *If you are in the Acquisition Program Office (APO):*
    - You might have to provide data to an Independent Review Team (IRT) conducting a Technology Readiness Assessment (TRA)
  - *If you are a Contractor:*
    - You might want to gain insight into how your proposals are evaluated
  - *If you are in The Aerospace Corporation:*
    - You might be invited to become a member of an IRT
- What is my objective in preparing this course?
  - *Guidance on this topic will always be inadequate due to the disruptive nature of technology innovation. I want to encourage you to understand the underlying, core principles rather than just trying to follow the letter of the DOD Deskbook.*



# Technology Readiness Assessments – the 64,000-foot View



- Public Law 109-163-Jan.6, 2006, Section 801

## **TITLE VIII—ACQUISITION POLICY, ACQUISITION MANAGEMENT, AND RELATED MATTERS**

*Subtitle A—Provisions Relating to Major Defense Acquisition Programs*

***SEC. 801. REQUIREMENT FOR CERTIFICATION BEFORE MAJOR DEFENSE ACQUISITION PROGRAM MAY PROCEED TO MILESTONE B.***

*(a) CERTIFICATION REQUIREMENT.—Chapter 139 of title 10, United States Code, is amended by inserting after section 2366 the following new section:*

***“§ 2366a. Major defense acquisition programs: certification required before Milestone B or Key Decision Point B approval***

*(a) CERTIFICATION.—A major defense acquisition program may not receive Milestone B approval, or Key Decision Point B approval in the case of a space program, until the milestone decision authority certifies that—*

***(1) the technology in the program has been demonstrated in a relevant environment; ...”***

---

Note that the term “**Key Decision Point**” is not in use since the cancellation of NSSAP 03-01



# Technology Readiness Assessments – the 64,000-foot View (Cont.)



- November 2, 2007 Air Force Memorandum on Technology Certification
  - *Spells out that for all Critical Technology Elements (CTEs) it has to be demonstrated in a relevant environment that they are at Technology Readiness Level (TRL) 6 or greater.*
- New provisions in the Weapon Systems Acquisition Reform Act of 2009

## TITLE I—ACQUISITION ORGANIZATION

***SEC. 104. Assessment of technological maturity of critical technologies of major defense acquisition programs by the Director of Defense Research and Engineering.***

***(a) ASSESSMENT BY DIRECTOR OF DEFENSE RESEARCH AND ENGINEERING.—***

***(1) IN GENERAL.*** — Section 139a of title 10, United States Code, is amended by adding at the end the following new subsection:

***(c) (1) The Director of Defense Research and Engineering, in consultation with the Director of Developmental Test and Evaluation, shall periodically review and assess the technological maturity and integration risk of critical technologies of the major defense acquisition programs ...***

***(2) The Director shall submit to the Secretary of Defense and the congressional defense committees by March 1 of each year a report...***

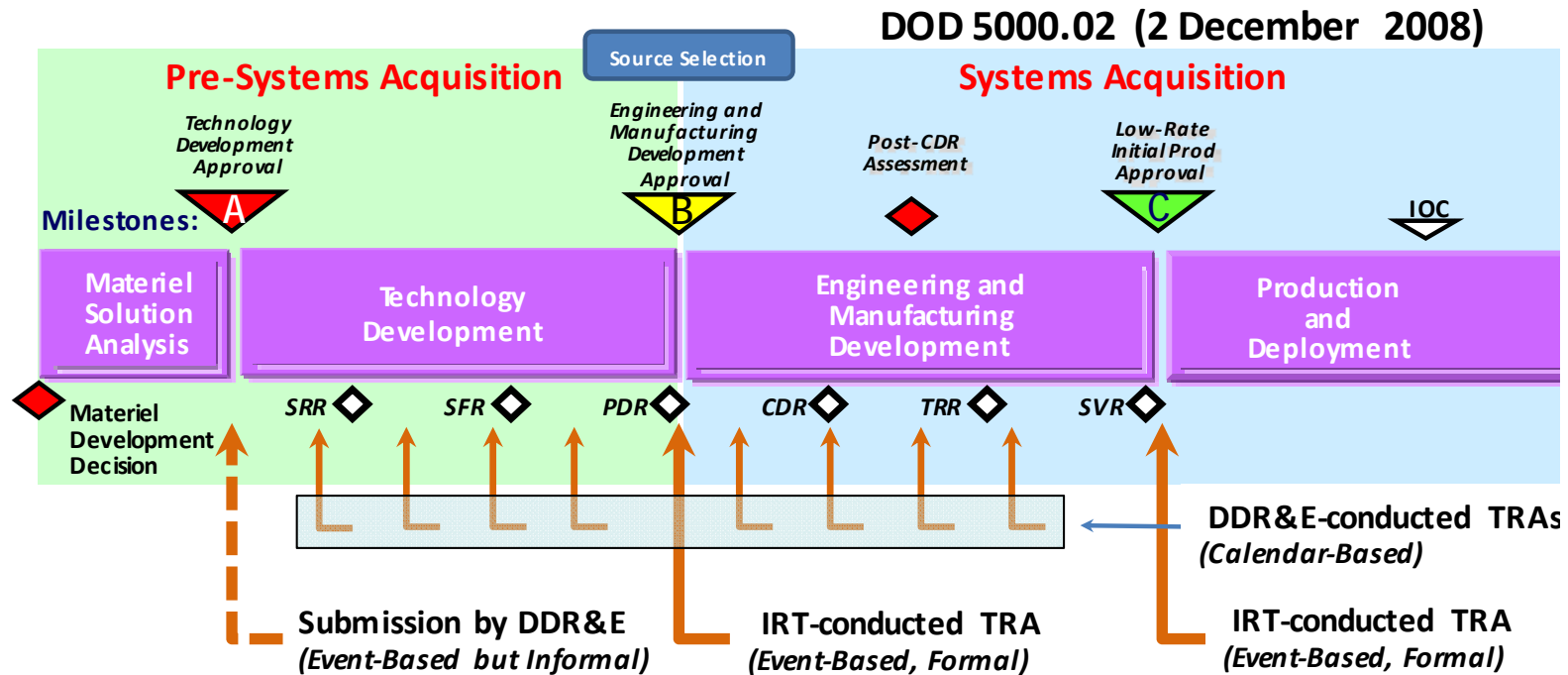
# Basic Department of Defense (DOD) TRA Definitions

- The key document providing DOD guidance on carrying out a TRA is entitled the “Technology Readiness Assessment (TRA) Deskbook”
  - *This tutorial is based on the most recent, July 2009 edition*
- Technology Maturity
  - *A measure or degree to which proposed technologies meet program objectives*
- Technology Readiness Assessment
  - *A TRA is a formal, systematic, metrics-based process and accompanying report that assesses the maturity of critical hardware and software technologies to be used in systems. The TRA is not intended to predict future performance of the evaluated technologies, nor does it assess the quality of the system architecture, design, or integration plan*
- TRA is different from “Conventional” Risk Management
  - *The result of a TRA is a single number on a 1-9, ordinal scale, called Technology Readiness Level (TRL).*
  - *TRLs do not intend to reflect either the likelihood of attaining required maturity or the impact of not achieving the required maturity*
- The TRA complements – but does not in any way preclude – the Program Manager’s responsibility to pursue reduction of all risks

# Critical Technology Elements

- Context for Technology Readiness Assessments
  - *For practical purposes not all planned technologies are assessed*
    - The technologies that are subject of a TRA will be called Critical Technology Elements (CTEs)
  - *However, the analysis of candidate technologies begins even before Materiel Development Decision takes place for the acquisition*
- A technology element is critical if
  - The system being acquired depends on this technology element to meet operational requirements within acceptable cost and schedule limits, **and**
  - The technology element or its application is
    - *either new or novel, **or***
    - *in an area that poses major technological risk during detailed design or demonstration*
  - *Candidate CTEs vs. CTEs*
    - Until it is approved by the Milestone Decision Authority (MDA,) all CTEs are considered only as Candidate CTEs
  - *CTE identification data includes the criteria and rationale for declaring the CTE as critical **or** eliminating it as a CTE candidate*

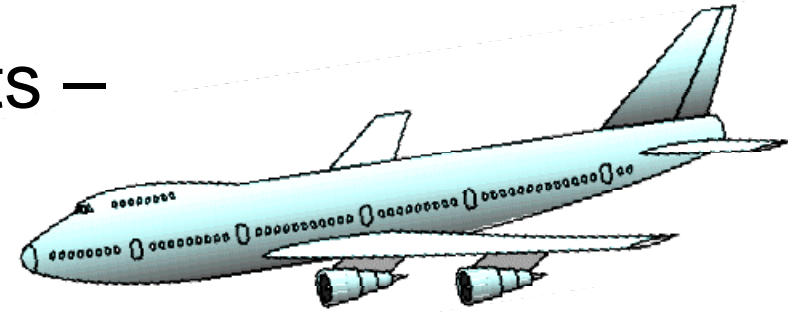
# Technology Evaluation Logistics



## Steps of a formal, IRT-conducted TRA at Milestones B and C

- *The Component Science & Technology Executive appoints an IRT of appropriate Subject Matter Experts (SMEs)*
- *Acquisition Program Office presents its technology plans to the IRT*
- *IRT evaluates the plan and submits the list of selected CTEs to the Defense Acquisition Board (DAB) for approval*
- *IRT assesses the maturity of the approved CTEs*
- *IRT briefs the Milestone Decision Authority (MDA) on its findings*
- *MDA approves/disapproves the entry to the next acquisition phase*

# Technology Readiness Assessments – the 10,000-foot View

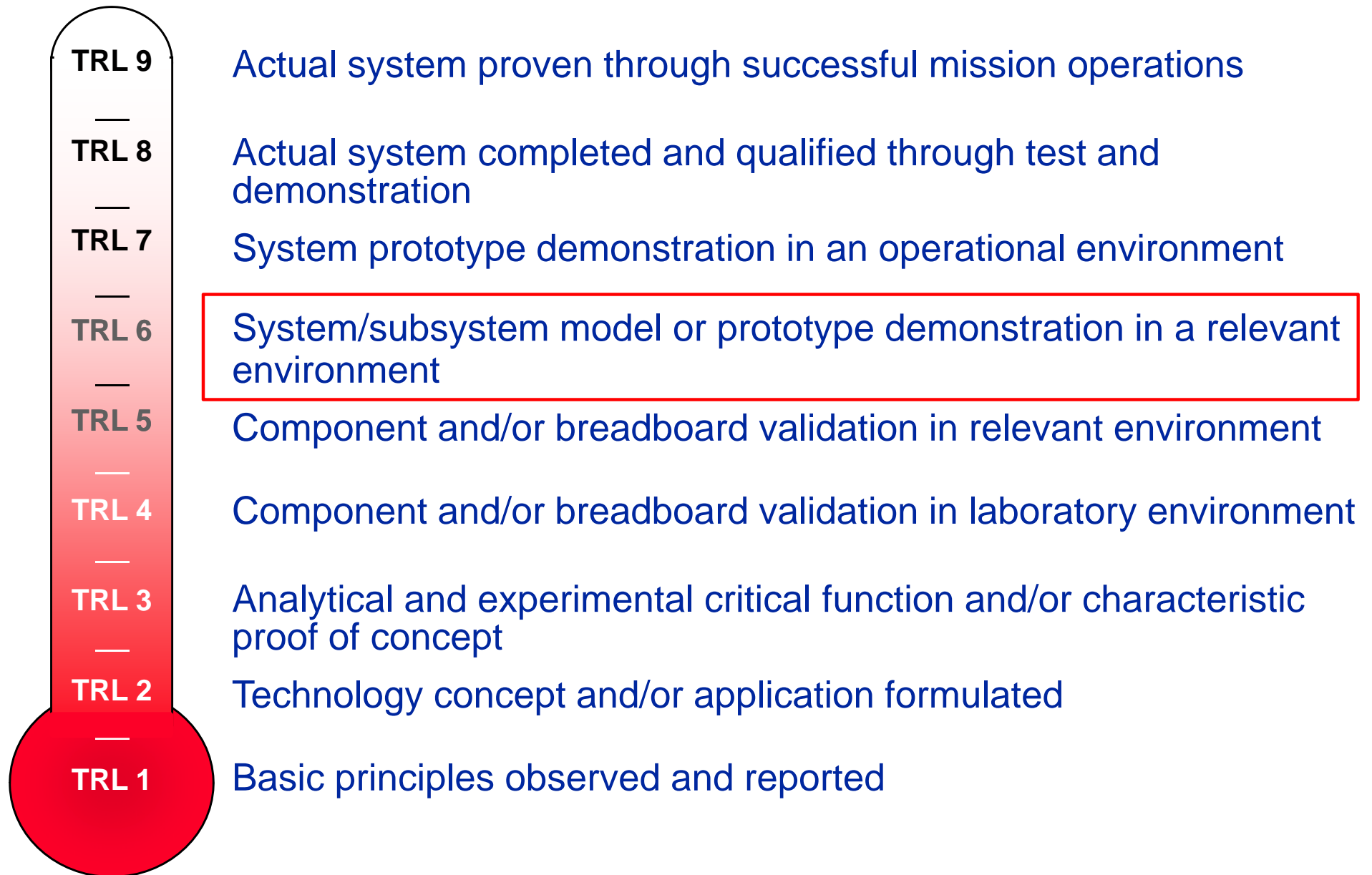


- Relevant Environment Definition
  - *Relevant Environment is a validation environment that simulates key aspects of the Operational Environment*
  - *The purpose of using a relevant environment is to demonstrate sufficient confidence in the CTE; i.e., that skillful application of this technology will fully support the required threshold functionality.*
- Relevant Environment for Space\*
  - *A satellite from launch to standard operation in space is exposed to drastically changing environmental conditions and a relevant environment test design must encompass all such stressing, aggregate conditions:*
    - Space Environment
    - Launch Environment
    - Designed Environment → This is where software “lives”
    - Operational Environment

---

*\*This is an experience-based recommendation; unfortunately the DOD Deskbook does not have adequate space-related guidance. For further details please see the backup slides.*

# Assessing CTEs using the TRL “Thermometer”



# Before We Move On...



- Check your understanding of the following new terms

- *IRT*



- *TRA*



- *TRL*



- *CTE*



- *Relevant Environment*



- O.K.?



# Software Technology Readiness Definitions\*



- The Definition of Software Technology
  - *Software technology is defined as the theory and practice of various sciences applied to software development, operation, understanding, and maintenance. Software Technology is any concept, process, method, algorithm, or tool whose primary purpose is the development, operation, understanding, and maintenance of software [Foreman 1997]*
  - *Software technology examples*
    - Technology directly used in the objective system
      - *E.g., two-tier and three-tier architecture, Service-Oriented Architecture (SOA,) Remote Procedure Calls (RPC)*
    - Technology used in tools that produce or maintain software
      - *E.g., Graphical User Interface (GUI) builders, programming languages and compilers, cyclomatic complexity analyzers*
    - Process technologies applied to produce or maintain software
      - *Personal Software Process (PSP), Cleanroom Software Engineering.*

---

\* Unfortunately, the TRA Deskbook does not have a definition of software technology

# A Sampler of Emerging Software Technologies to Watch

- Technologies directly supporting mission requirements
  - *Network Centric Warfare (NCW)*
    - NCW is a state-of-the art war-fighting theory with the following, two implementation dimensions
      - *Network Centric Operations (NCO), dealing with the cognitive and social dimensions of NCW*
      - *Network Centric Infrastructure (NCI), addressing physical and information dimensions of NCW*
    - Note that NCW almost automatically puts every weapon system in a System of Systems (SOS) context
  - *Service-Oriented Architecture (SOA)*
    - SOA is an emerging architecture style that may be used to implement Network Centered Infrastructure (NCI). Note that NCI is strongly promoted\* by the Office of the Undersecretary of Defense for Acquisition, Technology, and Logistics (OUSD(AT&L))

---

\* Source: [OUSD 2008], also see “Net Ready” as a standard Key Performance Parameter (KPP) in [DOD 2008]

# Emerging Software Technologies to Watch (cont.)

- Software technology breakthroughs exploiting advancements in hardware research and development
  - *Software for polymorphic computing*
    - These are hardware architectures that adapt to a particular objective
  - *Multi-threading*
    - True concurrent execution of threads on multiprocessors
  - *Software designed for low power consumption*
    - There are many possible code sequences that accomplish the same task; the objective of this research is to find code sequences that consume the least power and energy
- Software process technologies
  - *Model Driven Development (MDD)*
  - *Real-time garbage collection; real-time Java*
    - Enables the development of small footprint, high assurance software
  - *Incremental Commitment Process\**
  - *Agile Development*

---

\* For details on this new process see [Pew 2007]

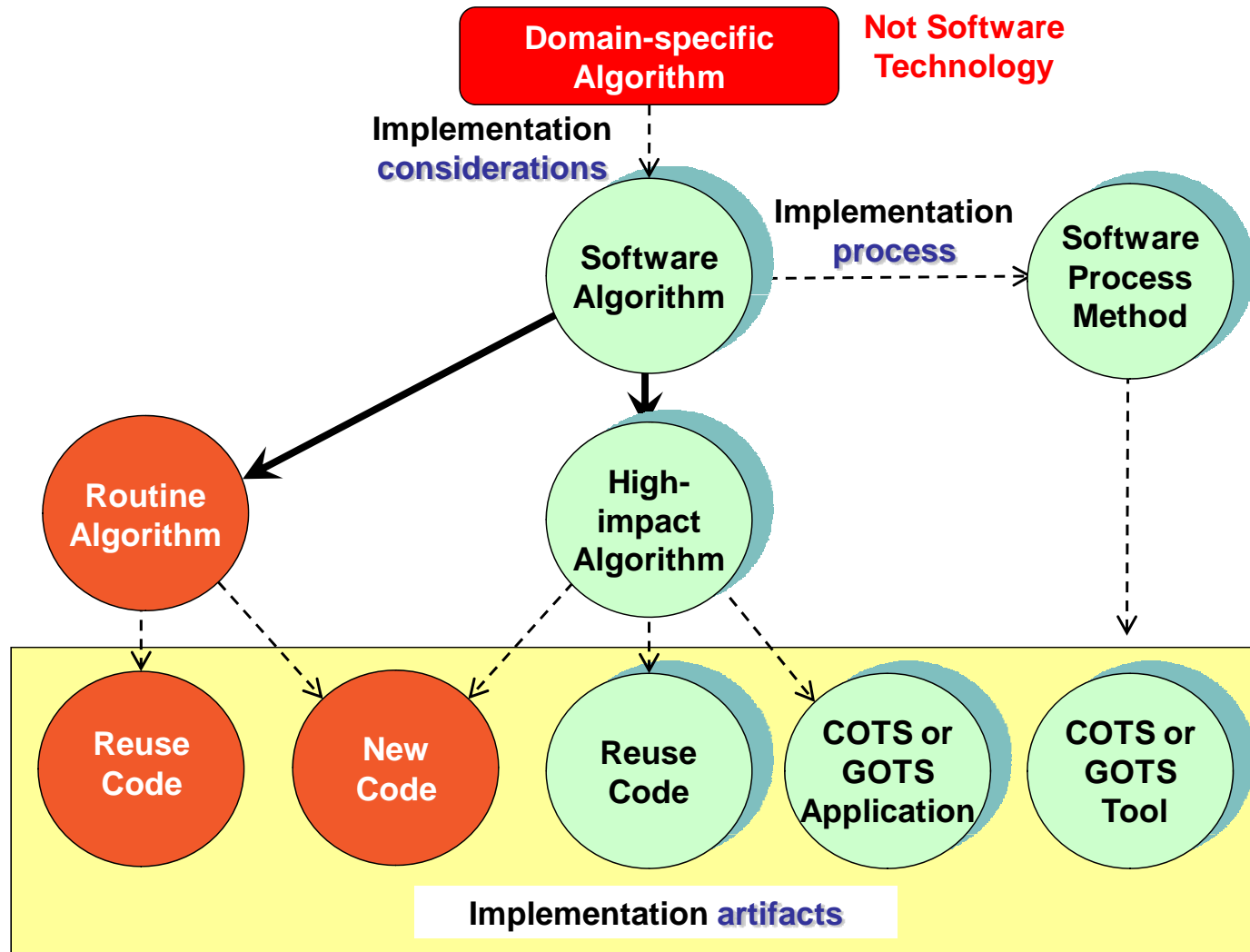
# Algorithms

- Basic, conventional definition
  - *An algorithm is a sequence of finite instructions. It is formally a type of effective method in which a list of well-defined instructions will, when given an initial state, proceed through a well-defined series of successive states, eventually terminating in an end-state.*
- Classification of algorithms from a TRA perspective\*
  - *Domain-specific algorithms*
    - Domain-specific algorithms implement various tasks in the user's domain
  - *Software algorithms*
    - Software algorithms implement various tasks in the software development domain
    - Implementation variations for software algorithms
      - *New code*
      - *Reuse code*
      - *Commercial Off-the-Shelf (COTS) and Government Off-the-Shelf (GOTS) software*

---

*\*Some source of confusion is that domain-specific algorithms might be implemented in hardware, firmware, or software (See filter example later.) Also, domain-specific algorithms are often tested with software tools, but that does not make them software algorithms.*

# What is In-scope for a Software TRA?

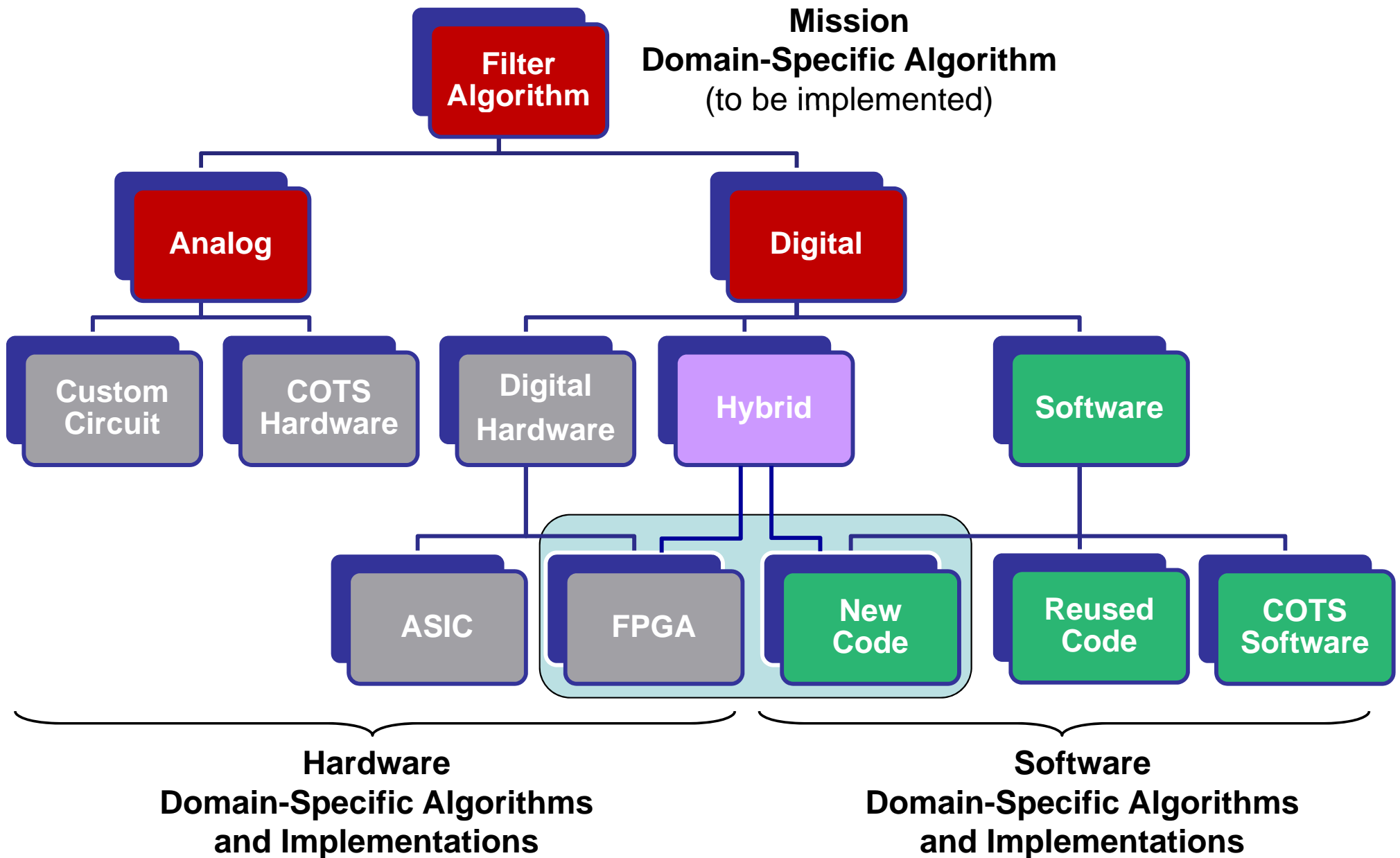


Legend: ● Red-colored items are never Software CTE candidates

# Algorithm Example: Filters

- Definitions
  - *Filters in Signal Processing*
    - A filter is a mathematical algorithm to remove part(s) of a signal
      - *In most cases the goal is to remove interfering noise to facilitate easier processing of the objective signal*
  - *Analog filter*
    - Analog filters are analog electronic circuit implementations of filter algorithms, operating on continuous-time, analog signals
  - *Digital filter*
    - Digital filters are digital electronic circuit or software implementations of filter algorithms, operating on sampled, discrete-time, digital signals
  - *Application-specific Integrated Circuits (ASICs)*
    - ASICs are customized integrated circuits for a particular application rather than intended for general-purpose use
  - *Field Programmable Gate Arrays (FPGAs)*
    - An FPGA is a semiconductor device that can be configured by the customer. FPGAs contain programmable logic components called logic blocks, and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together" like a one-chip programmable breadboard.

# Taxonomy of Filter Implementations





# Environmental Considerations

- Environmental categories according to the Deskbook:
  - *External or imposed environment*
    - Related to the operation of the product, may be either natural or man-made
  - *Internal or designed environment*
    - Always man-made, related to the designed product
- Further environmental dimensions
  - *Physical environment (for software, it is the designed environment)*
  - *Logical environment*
  - *Data environment*
  - *Security environment*
  - *User and use environment*
- Recommendation:
  - *Base software environmental analysis on the software architecture*

# What Can We Learn From the Software Architecture?\*

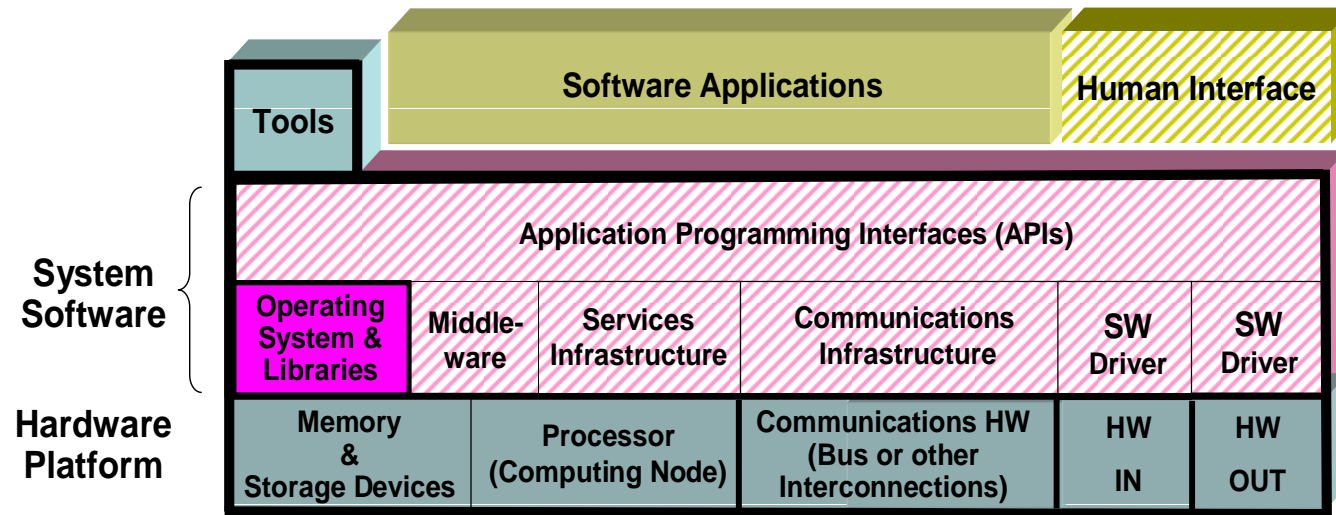
- Structural Viewpoint
  - *Elements, components of the system*
  - *Interactions between these components (“connectors”)*
  - *Structural organization of elements*
- Behavioral Viewpoint
  - *Dynamic actions of and within the system*
  - *Actions produced by the system*
  - *The ordering and synchronization of these actions*
  - *The behavior of system components and their interactions*
- Physical Interconnect Viewpoint
  - *Physical communication interconnects among system components*
  - *Layering among system components*
  - *Feasibility of construction, compliance with standards, and evolvability*

Note that all this information is relevant to technology selection and evaluation but even an elaborate Work Breakdown Structure (WBS) would not show it

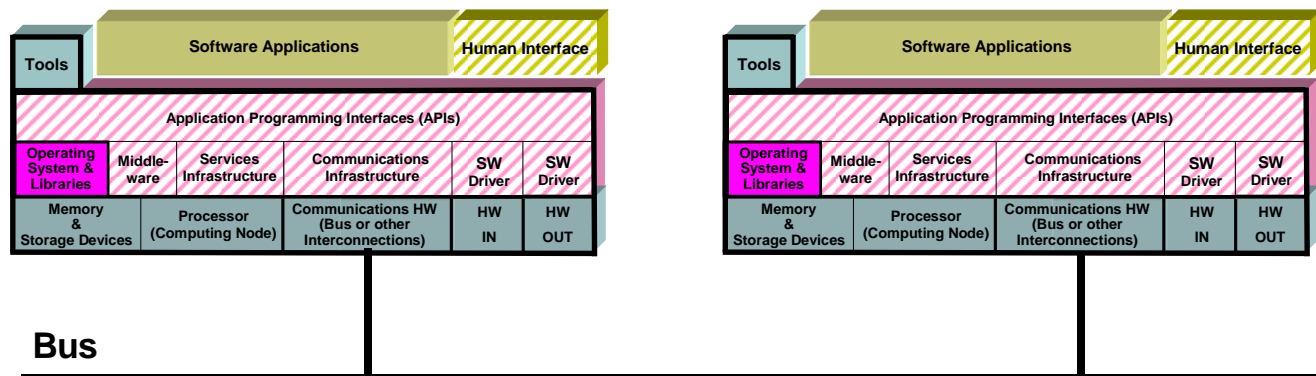
---

\* Discussion is based on the prevailing ISO/IEC architecture standard [ISO/IEC 2007]

# Selected, High-Level Viewpoints for TRA



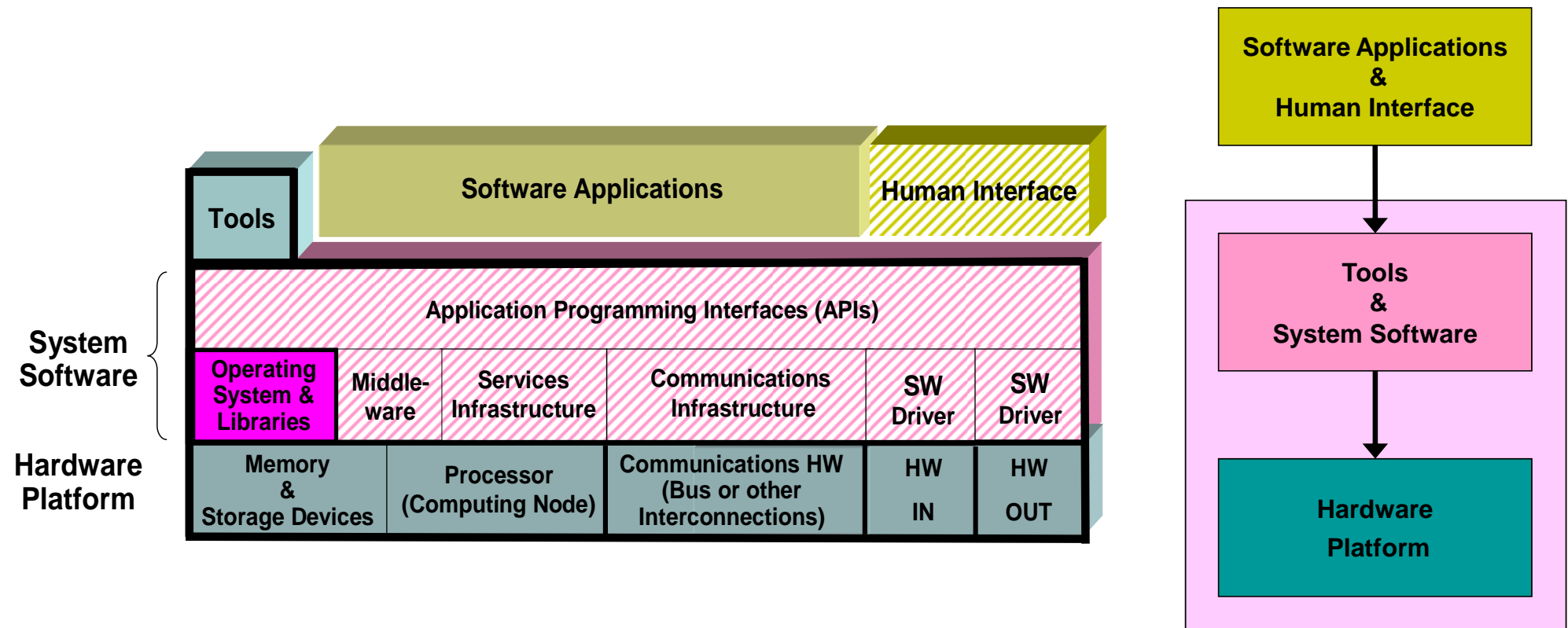
Structural Viewpoint: Software Deployed on a Single Node



Physical Interconnect Viewpoint: Software Deployed on Multiple Nodes

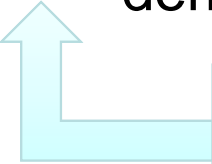
*\* Note that the diagrams are intentionally general to cover any kind of system, including computers on airplanes and ships. Space customization is discussed in the Case Studies.*

# Dependency of Software Environment Layers



# CTE Identification Questions\*

- 1) Does the technology have a significant impact on an operational requirement, cost, or schedule?
- 2) Does this technology pose a major development or demonstration risk?
- 3) Is the technology new or novel?
- 4) Has the technology been modified from prior successful use?
- 5) Has the software technology been repackaged such that a new relevant environment is applicable?
- 6) Is the technology expected to operate in an environment and/or achieve a performance beyond its original design intention or demonstrated capability?



A technology element is critical if the answer to the first question and to any of the remaining questions is “yes”

\* Source: DOD TRA Deskbook, pp B-8

# Disciplines and Knowledge Involved in TRL Determination

| # | Basic TRL DEFINITIONS from DOD TRA Deskbook*                                         | TRL GOALS                           | Knowledge Involved in Achieving HW Objectives | Knowledge Involved in Achieving SW Objectives | Overarching Systems Engineering Responsibilities |
|---|--------------------------------------------------------------------------------------|-------------------------------------|-----------------------------------------------|-----------------------------------------------|--------------------------------------------------|
| 1 | Basic principles observed and reported                                               | Demonstrate scientific feasibility  | Natural Sciences                              | Computer Science                              | N/A                                              |
| 2 | Technology concept and/or application formulated                                     |                                     |                                               |                                               |                                                  |
| 3 | Analytical and experimental critical function and/or characteristic proof of concept |                                     |                                               |                                               |                                                  |
| 4 | Component and/or breadboard validation in a laboratory environment                   | Demonstrate engineering feasibility | Hardware Engineering<br>Systems Engineering   | Software Engineering<br>Systems Engineering   | In-domain integration<br>Cross-domain evaluation |
| 5 | Component and/or breadboard validation in a <b>relevant environment</b>              |                                     |                                               |                                               |                                                  |
| 6 | System/subsystem model or prototype demonstration in a <b>relevant environment</b>   |                                     |                                               |                                               |                                                  |
| 7 | System prototype demonstration in an operational environment                         | Demonstrate operational feasibility | Hardware Engineering<br>Systems Engineering   | Software Engineering<br>Systems Engineering   | Cross-domain integration                         |
| 8 | Actual system completed and qualified through test and demonstration                 |                                     |                                               |                                               |                                                  |
| 9 | Actual system proven through successful mission operations                           | Demonstrate operations              | Mission Domain                                | Mission Domain                                | Mission Domain Demonstration                     |

\* Note that the basic definitions and goals for software and hardware are the same

# Software Technology Readiness Determination

- All Software TRLs are to be evaluated on the same, 7 dimensions
  - *These dimensions are recognized drivers of technology maturity demonstration. The objective is to track a maturation signature via evolution in these dimensions*
  - *A TRL is declared to be achieved if objective evidence has been provided that the status of specific conditions supports the satisfaction of the TRL goal and definition as stated*
- Software TRL evaluation dimensions
  - *Artifacts*
  - *Structural Context*
  - *Software Environment*
  - *Validation Environment and Methods*
  - *Data used for Validation*
  - *Configuration Management*
  - *Documentation*

---

*See evaluation details on the following slides*



# Evaluating Artifacts

| TRL | Basic TRL DEFINITIONS                                                                | Artifacts                                                                                                                                                                                     |
|-----|--------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | Basic principles observed and reported                                               | Research articles, peer-reviewed white papers, point papers, early conceptual models                                                                                                          |
| 2   | Technology concept and/or application formulated                                     | Analytic studies, papers on competing technologies                                                                                                                                            |
| 3   | Analytical and experimental critical function and/or characteristic proof of concept | Analytical and simulation models;<br>Availability of appropriate COTS/GOTS or reusable software artifacts is explored.                                                                        |
| 4   | Component and/or breadboard validation in a laboratory environment                   | A stand-alone prototype solving a partial-scale problem; Proposals for a nominal software architecture and for a simulation/stimulation work-up plan; COTS/GOTS, or reusable SW if applicable |
| 5   | Component and/or breadboard validation in a relevant environment                     | A stand-alone prototype solving a partial-scale problem; Detailed software architecture and final simulation/stimulation work-up plan; COTS/GOTS, or reusable SW if applicable                |
| 6   | System/subsystem model or prototype demonstration in a relevant environment          | Viable prototype providing the foundation for productization                                                                                                                                  |
| 7   | System prototype demonstration in an operational environment                         | Productized component                                                                                                                                                                         |
| 8   | Actual system completed and qualified through test and demonstration                 | Productized component                                                                                                                                                                         |
| 9   | Actual system proven through successful mission operations                           | Productized component                                                                                                                                                                         |

These tables are structured to help tracking the maturation signatures

# Evaluating the Structural Context and SW Environment

| TRL | Basic TRL DEFINITIONS                                                                | Structural Context     | SW Environment           |
|-----|--------------------------------------------------------------------------------------|------------------------|--------------------------|
| 1   | Basic principles observed and Reported                                               | n/a                    | "Academic", experimental |
| 2   | Technology concept and/or application formulated                                     | n/a                    | "Academic", experimental |
| 3   | Analytical and experimental critical function and/or characteristic proof of concept | n/a                    | "Academic", experimental |
| 4   | Component and/or breadboard validation in a laboratory environment                   | Prototype SW Component | "Academic", experimental |
| 5   | Component and/or breadboard validation in a relevant environment                     | Prototype SW Component | Operational-like         |
| 6   | System/subsystem model or prototype demonstration in a relevant environment          | Prototype WBS Level 3  | Operational-like         |
| 7   | System prototype demonstration in an operational environment                         | WBS Level 2            | Actual SW Environment    |
| 8   | Actual system completed and qualified through test and demonstration                 | WBS Level 1            | Actual SW Environment    |
| 9   | Actual system proven through successful mission operations                           | WBS Level 1            | Actual SW Environment    |

# Evaluation of the Validation Environment and Data

| TRL | Basic TRL DEFINITIONS                                                                | Validation Environment and Methods                                           | Data Used for Validation                                  |
|-----|--------------------------------------------------------------------------------------|------------------------------------------------------------------------------|-----------------------------------------------------------|
| 1   | Basic principles observed and Reported                                               | Basic research using analytical methods                                      | n/a                                                       |
| 2   | Technology concept and/or application formulated                                     | Applied research using analytical methods                                    | Synthetic data only                                       |
| 3   | Analytical and experimental critical function and/or characteristic proof of concept | Active R&D initiated via the use of models and simulation                    | Partially representative data                             |
| 4   | Component and/or breadboard validation in a laboratory environment                   | Advanced technology development with throwaway or evolutionary SW prototypes | Representative data                                       |
| 5   | Component and/or breadboard validation in a relevant environment                     | Advanced technology development with throwaway or evolutionary SW prototypes | Representative data                                       |
| 6   | System/subsystem model or prototype demonstration in a relevant environment          | Development using evolutionary SW Prototype                                  | High-fidelity data representative of relevant environment |
| 7   | System prototype demonstration in an operational environment                         | End-to-end testing of Production SW using system simulator                   | High-fidelity data representative of relevant environment |
| 8   | Actual system completed and qualified through test and demonstration                 | Testing of Production SW using OT&E                                          | Real data                                                 |
| 9   | Actual system proven through successful mission operations                           | Actual Mission                                                               | Real data                                                 |

# Evaluation of CM and Documentation

| TRL | Basic TRL DEFINITIONS                                                                | Configuration Management                                 | Documentation                                                                                                                                                |
|-----|--------------------------------------------------------------------------------------|----------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | Basic principles observed and reported                                               | n/a                                                      | Appropriate and sufficient to demonstrate basic principles                                                                                                   |
| 2   | Technology concept and/or application formulated                                     | n/a                                                      | Appropriate and sufficient to demonstrate application concept                                                                                                |
| 3   | Analytical and experimental critical function and/or characteristic proof of concept | n/a                                                      | Appropriate and sufficient to interpret analytical or experimental data; Full documentation available on COTS/GOTS or reusable software under consideration. |
| 4   | Component and/or breadboard validation in a laboratory environment                   | Limited scope; appropriate for experimental environment  | Appropriate and sufficient to interpret experimental results; Full documentation available on chosen COTS/GOTS or reusable software                          |
| 5   | Component and/or breadboard validation in a relevant environment                     | Appropriate for operational-like, production environment | Appropriate and sufficient to interpret results; Full documentation on chosen COTS/GOTS or reusable software                                                 |
| 6   | System/subsystem model or prototype demonstration in a relevant environment          | Appropriate for operational-like, production environment | Appropriate and sufficient to validate relevant environment and interpret demonstration results                                                              |
| 7   | System prototype demonstration in an operational environment                         | Appropriate for the actual environment                   | Appropriate and sufficient to validate test results                                                                                                          |
| 8   | Actual system completed and qualified through test and demonstration                 | Appropriate for the actual environment                   | Appropriate and sufficient to validate technology's performance and to operate and maintain the product.                                                     |
| 9   | Actual system proven through successful mission operations                           | Consistent with the actual environment                   | Appropriate and sufficient to validate technology's performance and to operate and maintain the product.                                                     |

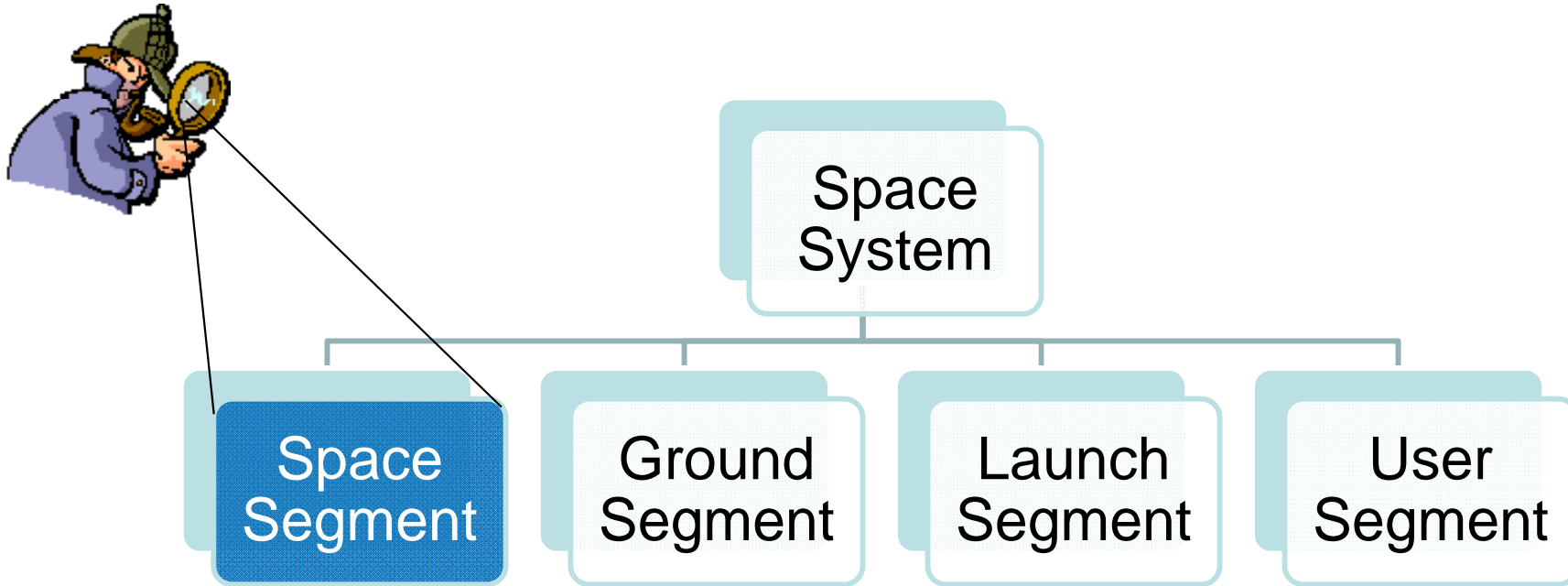
# Space Vehicle Case Study

**“In Space No One Can Hear You Scream”**

~~~ (Tagline of the 1979 Movie “Alien”)




# Software CTE identification for a Space Vehicle



- Assumptions and Constraints
  - *The example space vehicle is simplified and hypothetical*
  - *Only selected, “mission-neutral” satellite functions will be discussed*

# Architectural Documentation-related Definitions\*

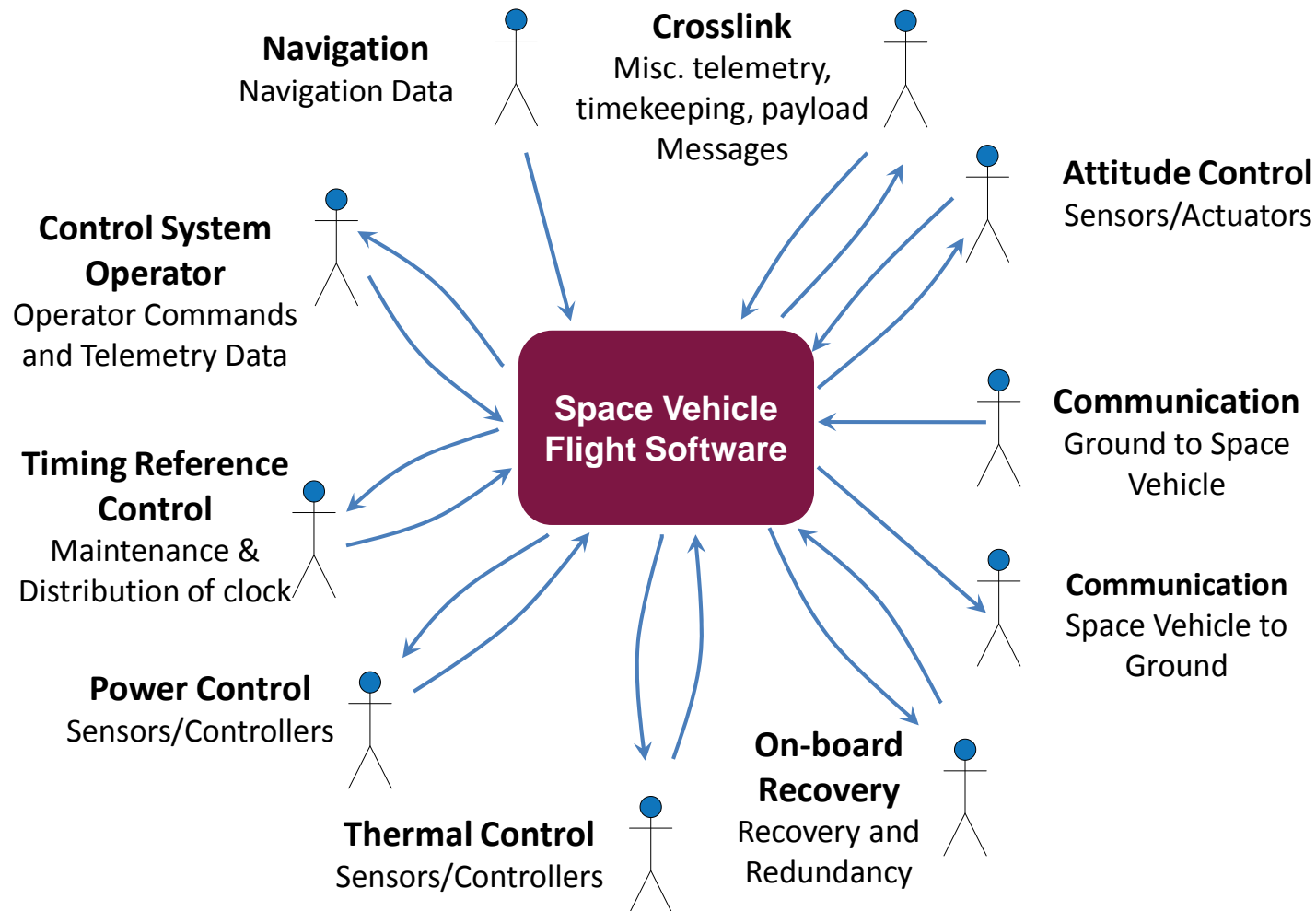
- Unified Modeling Language (UML®)
    - *UML is a language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system*
  - Context
    - *Context includes all those things that on the outside interact with the system*
  - Actor
    - *An Actor represents a role that a human, a hardware device, or another system plays with the objective system*
    - *The Actor symbol in UML is as follows:*
- 
- Deployment Diagram
    - *A diagram showing the nodes that form the system's hardware topology on which the system executes*

---

\* Source: [Booch 1999]

® UML is Registered in the U.S. Patent and Trademark Office by The OMG, Inc.

# Partial\* Space Vehicle Flight Software Context Diagram



\* Only selected, "mission-neutral" satellite functions are shown

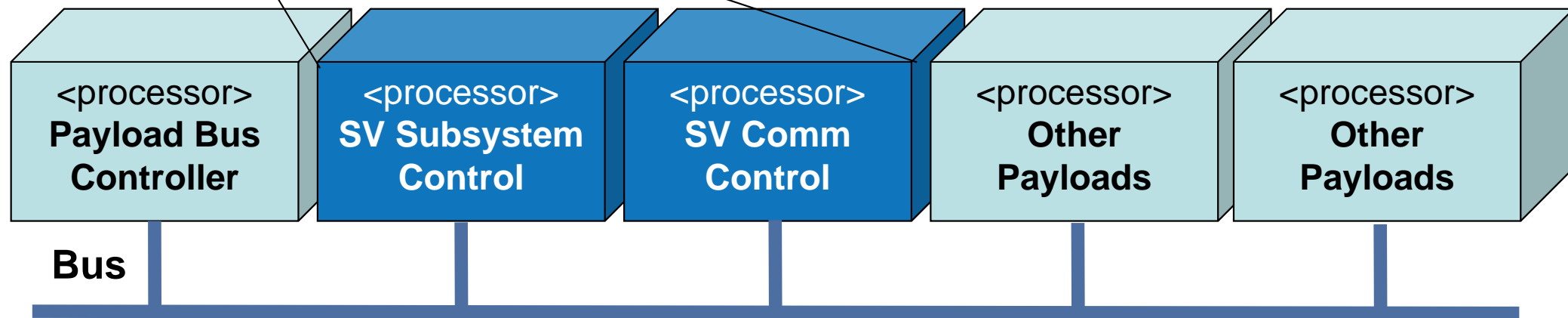


# Sample Worksheet for Software CTE Identification

| FUNCTIONAL AREA                                 | COMMENTS   | ARE THERE ANY SOFTWARE CTE CANDIDATES?* |
|---|--|---|
| Attitude Control                                | <ul style="list-style-type: none"> <li>• Reading attitude sensors</li> <li>• Control attitude actuators</li> </ul>               | Most likely no                          |
| Communication (External)                        | <ul style="list-style-type: none"> <li>• Space Vehicle to Ground</li> <li>• Ground to Space Vehicle</li> </ul>                   | Maybe<br>(Depends on mission)           |
| Communication (Internal)                        | <ul style="list-style-type: none"> <li>• Interfacing with other payloads</li> </ul>  | Most likely no                          |
| Crosslink Processing                            | <ul style="list-style-type: none"> <li>• Miscellaneous telemetry, timekeeping, and payload messages across satellites</li> </ul> | Most likely no                          |
| Data Base Management                            |  | Most likely no                          |
| Payload Bus Controller                          |  | Most likely no                          |
| Diagnostics and Maintenance                     | <ul style="list-style-type: none"> <li>• Recovery management</li> <li>• Redundancy processing</li> </ul>                         | Most likely no                          |
| Electrical Power Subsystem Control              | <ul style="list-style-type: none"> <li>• Sensors</li> <li>• Controllers</li> </ul>   | Most likely no                          |
| Keeping Time (Clock)                            | <ul style="list-style-type: none"> <li>• Maintaining time information</li> </ul>   | Most likely no                          |
| Navigation/Management of the Space Vehicle (SV) | <ul style="list-style-type: none"> <li>• Reading telemetry data</li> <li>• Commanding the SV</li> </ul>                          | Most likely no                          |
| Thermal Control                                 | <ul style="list-style-type: none"> <li>• Sensors</li> <li>• Controllers</li> </ul>   | Most likely no                          |
| etc.  |  |   |

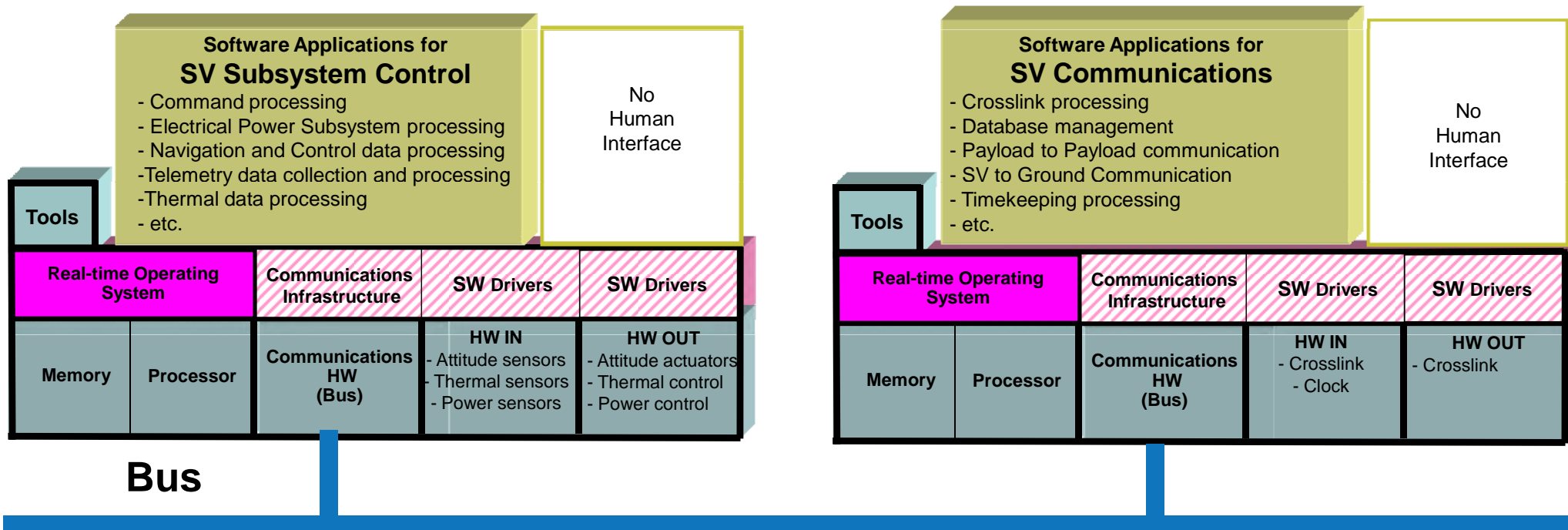
*\* Note that in case of any potential candidates the 6 CTE identification questions would have to be answered for final classification*

# UML Deployment Diagram for the Space Vehicle



- Payload Bus Controller is usually a COTS solution
- Other payloads could be miscellaneous communications, imagery, infra-red radar, etc. processing, depending on the actual mission

# TRA-Oriented, Customized Deployment Diagram\*



## • Conclusions:

- Apparently no need for “extreme” new or novel solutions
- For sake of simplicity we assume that no reuse or COTS are planned
- **We still need to examine dependency and tools issues**

\* Note that distributing functionality across processors is an architecting step. The deployed functionality on the processor does not always neatly match the subsystem name.

# Detour: Dealing with Radiation in Space Hardware

- Why is electromagnetic interference a problem?
  - *In presence of ionizing radiation a single charged particle can knock thousands of electrons loose*
- Major radiation damage sources in space
  - *Solar particle events*
  - *Van Allen radiation belts*
  - *Cosmic rays, etc.*
- Selected effects of radiation on electronics
  - *Rearrangement of the atoms, creating lasting damage*
  - *Transient ionization effect or latchup*
    - During a single-event latchup, heavy ions or high-energy protons are passing through the inner-transistors of the circuit, causing a “short”
- Radiation hardening (“rad-hard”)
  - *It is a method of designing and testing electronic components to make them resistant to electromagnetic radiation damage*

# Radiation Hardening

- Selected radiation hardening approaches\*
  - *Shielding*
    - Involves complete shielding of assemblies and subsystems
  - *Radiation-Hardening-by-Process (RHBP)*
    - This is the “classic” approach to rad-hard
    - “Process” refers to the semiconductor fabrication process
      - *Chips made on special, insulating substrates like silicon oxide or sapphire*
      - *Choice of substrate with wide band gap*
  - *Radiation-Hardening-by-Design (RHBD)*
    - The radiation mitigation techniques are implemented in layout or in the chip architecture and not in the fabrication process
  - *Augmenting reliability techniques*
    - Using redundant elements on circuit- and/or system level
    - Applying a watchdog timer to perform a hard reset if needed

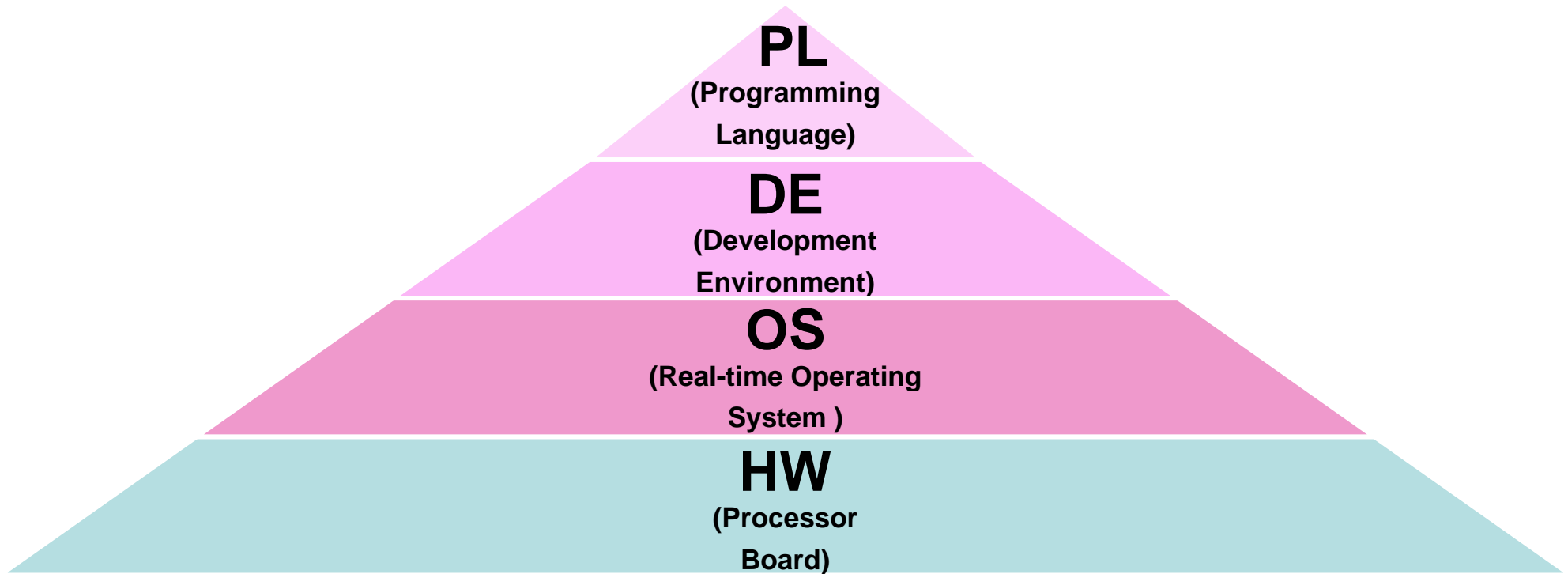
---

\* Reference: [Barnaby 2004]

# Software Technology Readiness Implications

- Shielding is the only solution without software implications
  - *However, it is bulky and heavy*
    - Remember, excess weight carries a special penalty in space designs
- In case of all other solutions exercise caution
  - *Many vendors developed rad-hard processors that are based on (and claimed to be equivalent with) various, commercial processors, such as the IBM PowerPC. However, they have to be evaluated on a case-by-case basis*
    - For real-time applications the development environment, including the used compiler, has to be validated on the same rad-hard platform
    - With respect to the objective system, the main concern is the preservation of real-time characteristics when the software, at the last phase of the development process, is moved from a commercial, developer platform to a rad-hard platform (“Operational-like” Software Environment)
- Conclusion
  - *The understanding of hardware technology options and their maturity is essential for a successful software TRA*

# The Dependency of Software TRLs on Hardware TRLs



$$\text{TRL(PL)} \leq \text{TRL(DE)} \leq \text{TRL(OS)} \leq \text{TRL(HW)}$$

*The equation above saves the effort of fully understanding the details of the particular layers ☺*

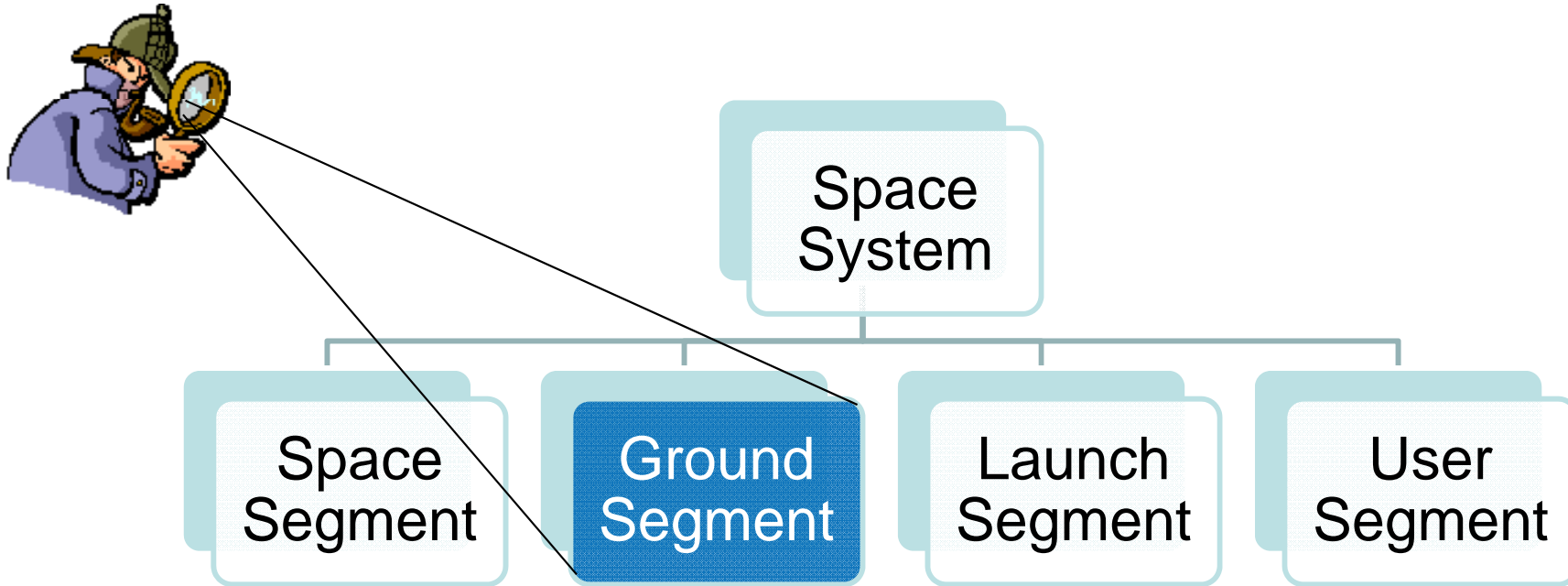
# Ground System Case Study

Is Service-Oriented Architecture (SOA) a Critical Technology Element?

“...And there is nothing new under the sun.  
Is there anything of which it may be said, ‘See, this *is* new’?  
It has already been in ancient times before us.”  
~~~ Ecclesiastes 1:9-10



# SOA as a CTE in a Ground System



- SOA as a CTE?
  - *Google produced 40 million (!) hits in 0.2 sec for “SOA”. Even if we discount hits on the Society of Actuaries and such, it is very impressive. Wouldn't it prove that it is a mature technology?*
  - *No. Using SOA is a risky proposition and extreme caution is needed. SOA belongs to the category where concepts and new code, reuse, and COTS dimensions all might have to be evaluated.*

# Detour: What is a Service-Oriented Architecture (SOA)?

- Architecture\*
  - *“Architecture is the fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.”*
- Service-Oriented Architecture\*\*
  - *“A Service-Oriented Architecture takes advantage of networking capabilities to integrate applications in a way that is independent of architecture, programming language, development platform and vendor. Through a set of standard interfaces, services are made available to any consumer willing to follow the rules for interface and consumption.”*
- Selected, generic SOA services
  - *Messaging, mediation/translation between data structures and protocols, Data Base Management System (DBMS,) high-speed networking, collaboration, Information Assurance/Security, etc.*
- Question to ponder
  - *What do you think the benefits of using such an architectural style are?*

---

\* Source: [ISO/IEC 2007]

\*\* Source: [Minkiewicz 2007]

# The Road to SOA for Space

- SOA is a promising approach to implement Operational Responsive Space (ORS) and Joint Warfighting Space (JWS)
- Operational Responsive Space
  - *ORS is characterized by an incremental approach from prototyping to production, on the basis of highly modularized capabilities*
  - *According to the early ORS ideas, the key to achieving these objectives is space system bus\* standardization*
    - Note that the term “bus” in ORS equally relates to all segments of a space system, not only to the space vehicle.
- Joint Warfighting Space [Schuler 2005]
  - *The JWS initiative seeks to make space an organic part of joint task forces in theater*
  - *ORS is an enabler of JWS*
- Question to ponder
  - *What do you think the connection is between ORS and the earlier mentioned NCW doctrine?*

---

\* ORS was originally introduced by the now defunct Office of Force Transformation (OFT) DOD entity. Here we only refer to the generic aspects of the earlier OFT proposal.

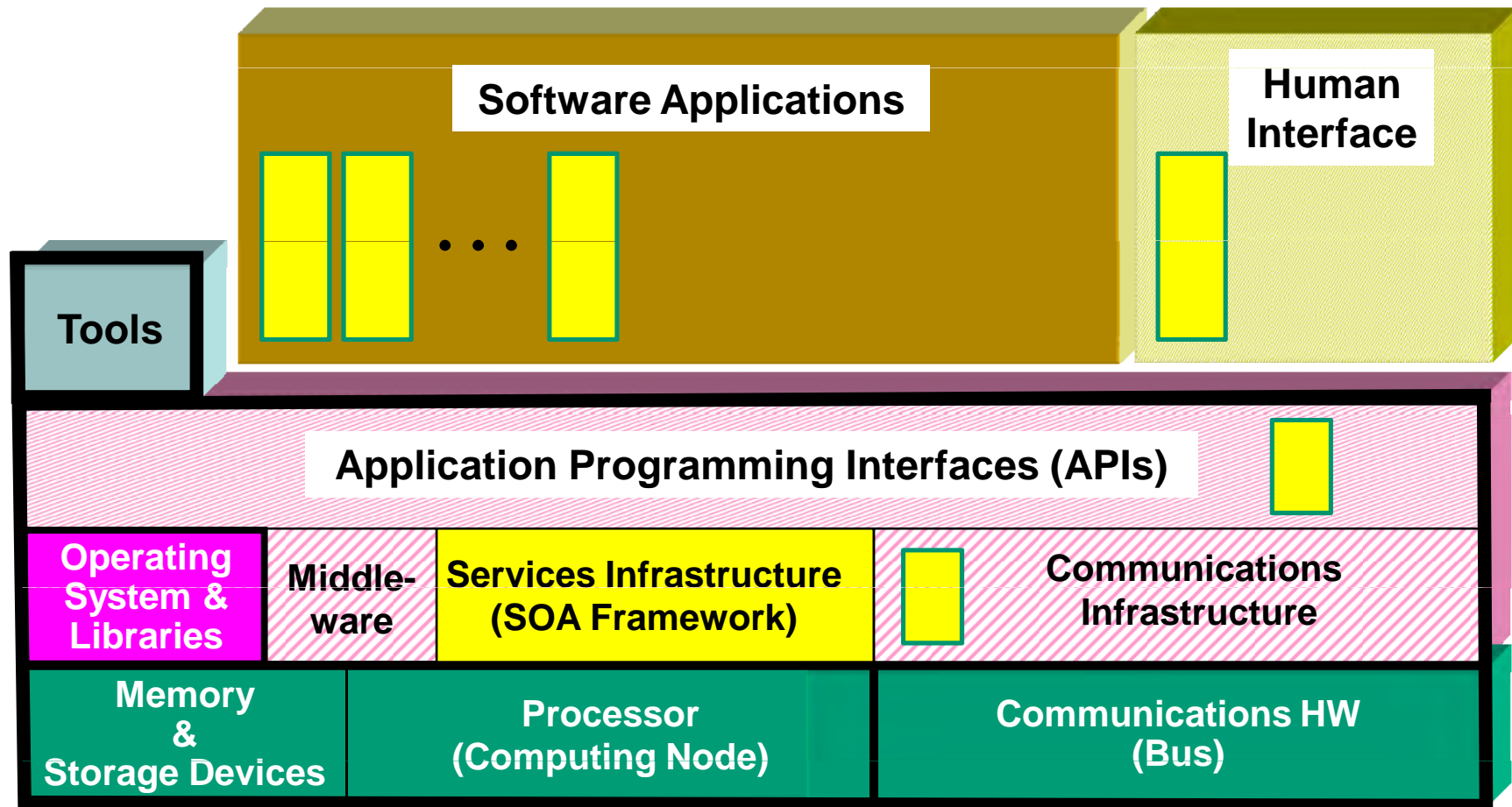
# Selected SOA Services for a Ground System

- Sample Ground functionality that could be implemented via services\*
  - *Command Processing*
    - Commands to space vehicles
    - Commands to antennae systems
  - *Orbital Data Processing*
  - *Critical alarms*
  - *Mission Planning*
  - *Real-time Telemetry Processing*
  - *Processing/providing data on external interfaces*
    - Other ground stations
    - External clients of ground station services
  - *Situational awareness*
  - *Etc.*
- Of course a SOA Framework would be needed as well
  - *E.g., to implement the registry that facilitates the seamless integration, upgrade, discovery and invocation of services*

---

\* *Caveat: SOA does not always make sense for implementing all Ground System functionality*

# SOA Components in a Ground System

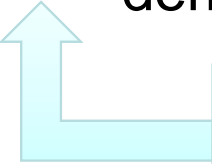


Legend:  Potential SOA Component

Note that this GS example does not have HW input/output elements other than the bus

# CTE Identification Questions Revisited

- 1) Does the technology have a significant impact on an operational requirement, cost, or schedule?
- 2) Does this technology pose a major development or demonstration risk?
- 3) Is the technology new or novel?
- 4) Has the technology been modified from prior successful use?
- 5) Has the software technology been repackaged such that a new relevant environment is applicable?
- 6) Is the technology expected to operate in an environment and/or achieve a performance beyond its original design intention or demonstrated capability?



A technology element is critical if the answer to the first question and to any of the remaining questions is “yes”

# CTE Identification Question 1

- (1) Does the technology have a significant impact on an operational requirement, cost, or schedule?
- Regarding operational requirements the answer is definitely **YES**, due to the Net-Ready KPP and the earlier mentioned OUSD SOA directive implications
  - Also, regarding Cost and Schedule, service-orientation provides numerous enablers for improvement
    - *The framework and core building blocks can be acquired as COTS*
    - *Services allow substantial reuse*
    - *A properly designed SOA framework enables the automatic discovery of fine-grained software services, negotiates their acquisition, and composes, binds, executes, and unbinds them*
    - *The use of SOA speeds-up the incremental delivery of new capabilities in the system evolution context*
    - *Through loose coupling and tight interface standards, consumers of services need only know how to interact with a service, and there is no need to understand deeper details*

## CTE Identification Questions 2-4

(2) Does this technology pose a major development or demonstration risk?

- **YES.** SOA is very pervasive, the associated components show up in many different parts of the overall system

(3) Is the technology new or novel?

- **NO.** Do you remember Distributed Object Architecture (DOA,) Common Object Model (COM,) Object Request Broker (ORB,) Common Object Request Broker Architecture (CORBA,) etc.?

(4) Has the technology been modified from prior successful use?

- It depends. However, most likely the answer is **NO**, because the SOA COTS elements would not be modified



# CTE Identification Questions 5-6

- (5) Has the technology been repackaged such that a new relevant environment is applicable?
- **Maybe.** “Repackaging” is a broad term and involves considerations for both the hardware and systems software platforms of the SOA services. This question is particularly critical when services are independently adopted for the objective system
- (6) Is the technology expected to operate in an environment and/or achieve a performance beyond its original design intention or demonstrated capability?
- **YES.** One of the main concerns with SOA implementations is the overhead associated with service invocation/execution and its impact on performance and Quality of Service (QoS). It is very unlikely that the structure of the objective system would be identical to a prior system already in use; hence these factors should be verified in advance

# Evaluation of Answers

- 1) Does the technology have a significant impact on an operational requirement, cost, or schedule? **YES**
- 2) Does this technology pose a major development or demonstration risk? **YES**
- 3) Is the technology new or novel? **NO**
- 4) Has the technology been modified from prior successful use? **NO**
- 5) Has the software technology been repackaged such that a new relevant environment is applicable? **Maybe**
- 6) Is the technology expected to operate in an environment and/or achieve a performance beyond its original design intention or demonstrated capability? **YES**

Conclusion: Due to YES answers to questions 1, 2, and 6, SOA is a CTE

Reminder: A technology element is critical if the answer to the first question and to any of the remaining questions is “yes”

# More SOA Specifics

- The 6 questions need to be carefully applied and special attention to be paid to the following specifics:
  - *Platform/Relevant Environment compatibility for COTS SOA elements*
    - Note that different services might come from different sources
  - *Inter-component Compatibility of acquired COTS SOA elements*
    - See reason given at the prior issue
  - *Execution performance considerations and scaling of services*
    - SOA involves a lot of overhead; as it was mentioned earlier, the impact needs to be carefully assessed in advance
  - *SOA interface adequacy for new services*
    - Service uniformity is achieved via standard interfaces; however, beyond the core functionality, the inherent throughput, data capacity, error tolerance, testability, etc. might not be enough for the new service.
  - *Feasibility and effort needed to interface reused and legacy software*
    - Due to the fact that most likely only selected functionality would be provided as a “service”

# COTS/Reuse Evaluation Revisited\*

- The application of COTS/Reused Software is a very attractive but at the same time very risky approach to software development
  - *Consequently, we need clarity on what COTS/Reuse Attributes are in-scope for a TRA and what inquiries belong to the “routine”, programmatic risk management area*
- In-scope for TRA
  - *Accuracy, correctness*
  - *Availability*
  - *Robustness*
  - *Security*
  - *Product performance*
  - *Testability*
  - *Version compatibility*
  - *Inter-component Compatibility*
  - *Functionality*
- In-scope for Trades and Risk Reduction
  - *Documentation quality*
  - *Ease of Use*
  - *Flexibility*
  - *Installation/Upgrade Ease*
  - *Portability*
  - *Price*
  - *Vendor support and maturity*
  - *Training*
  - *Vendor concessions*

---

\* Note that the assessment of evaluation, glue code writing, and integration cost (effort and schedule) is also out of scope for a TRA while it is a very critical planning activity.

# Challenging Topics

(What they Don't Teach at the Defense Acquisition University...)

**“Technological change is like an  
axe in the hands of a pathological  
criminal.”**

~~~ Albert Einstein

# “Challenging” is a Euphemism for Controversial...



- Sources of controversy
  - *Incorrect or ambiguous government guidance*
  - *Lack of consensus amongst the practitioners*
  - *Mismanaged expectations*
- Issues to be discussed
  - *The consequences of rolling-up TRLs*
    - The need to roll-up TRLs many times manifests itself as a call for providing a single system maturity index
      - *A sub-issue that needs to be addressed is the attempt for algebraic manipulation of an ordinal measurement scale*
  - *Putting too much faith into the predictive ability of TRLs*
    - The underlying issue is to understand that technology development should be viewed as an innovation process and as such it is highly unpredictable
  - *TRA vs. Risk Management*
    - We will look at a particular example, programming language selection for high assurance software, where they are in conflict
  - *The confusion between Technology Maturity and Software Maturity*
    - We will look at the reasons, i.e., confusing definitions and lack of understanding of the systems and software engineering life cycle processes

# Rolling-up TRLs

- Rolled-up TRLs are supposed to provide one single maturity number for a system or a SOS
  - *The idea equally relates to a single system with multiple CTEs or a SOS with dominant CTEs (or rolled-up TRLs...) assigned to the participating systems*
- Question: What would be your recommendation for a roll-up rule?

a) *Use the lowest TRL*

$$TRL_{sys} = \text{MIN} (TRL_1, TRL_2, \dots, TRL_n)$$

b) *Compute the arithmetic average of TRLs*

$$TRL_{sys} = \frac{TRL_1 + TRL_2 + \dots + TRL_n}{n}$$

c) *I have another method to do the roll-up*

d) *I don't recommend rolling-up TRLs*



# Rolling-up TRLs: My Answers in the order of preference

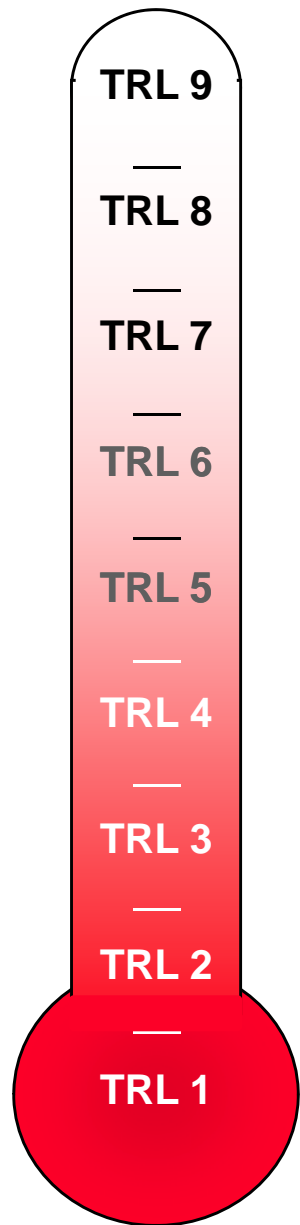
- **(d)** Don't roll-up TRLs
  - *No real need for roll-up*
    - If CTE selection has been correctly carried out then we should not have too many technology elements to deal with
  - *It is beneficial to keep the maturity evolution of CTEs separately visible*
    - However, in case of dependency they need to be collectively re-evaluated
- **(a)** Use the lowest TRL
  - *This approach is based on well-known reliability principles*
    - If all selected CTEs are indeed critical\* then the weakest link should be used to describe the maturity of the overall system or SOS
- **(b)** Arithmetic average of TRLs
  - *Not acceptable. One mustn't do arithmetic on ordinal scales under any circumstances*
- **(c)** Do you have any suggestions?

---

\* Remember, the 6 CTE identification questions should narrow down the candidate CTE list

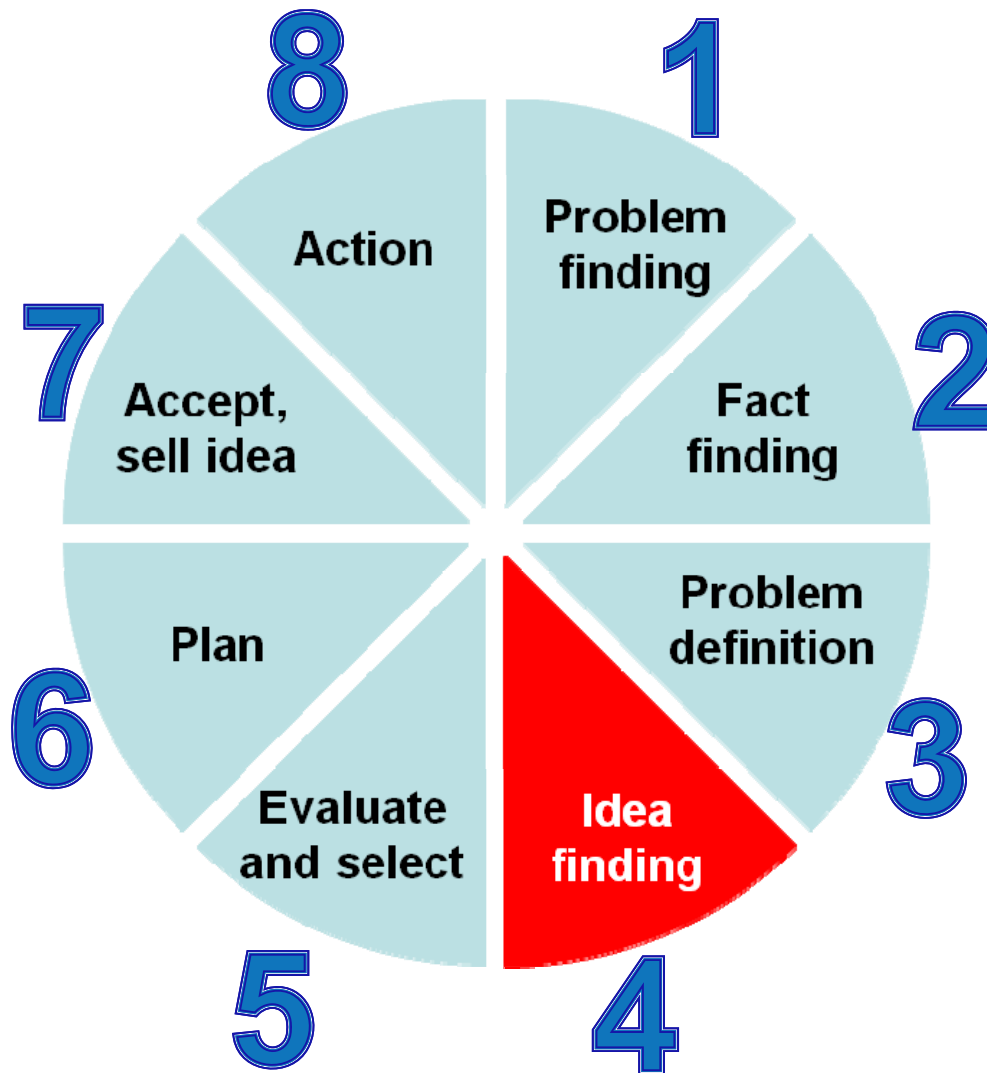


# TRL as an Ordinal Scale



- Unlike in “classic” risk prioritization schemes, a single, 1-9 integer is assigned to every maturity level
  - *For any CTE the impact of not moving-up from any level to the next within the program’s cost and schedule constraints is the loss of the total program*
- In conventional risk management
  - *Risks are characterized with their likelihood and their impact*
  - *Risk burn-down plans can be put in place to reduce or eliminate risks*
- What do the TRLs represent?
  - *TRLs represent a certain level of growing confidence that the technology will perform as intended. However, we cannot quantify this confidence beyond the “warm feeling” what an ordinal scale could provide*
    - We don’t know the likelihood of success at any levels
    - We cannot easily estimate the costs either
      - *Details to follow on the next few slides*

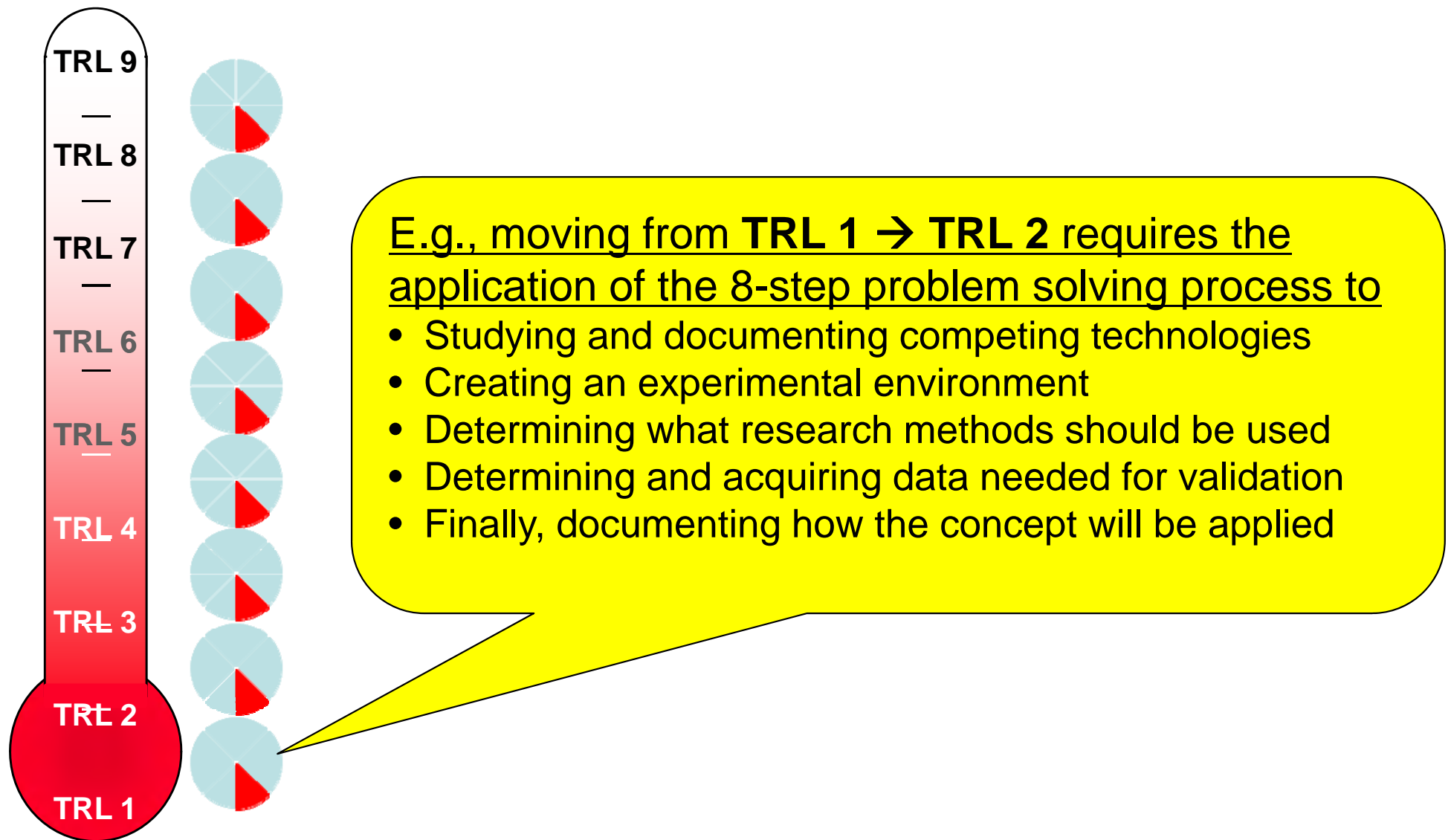
# A Creative Problem Solving Profile for Innovation\*



- Despite the diagram's neat symmetry, in reality the steps in the slices have different
  - Durations
  - Activity types
  - Levels of complexity
  - Levels of difficulty
- In fact, according to Basadur, the profile not only changes by problem type, but it is also different individual by individual
- **Phase 4 is the most volatile**

\* Source: [Basadur 2001]

# Technology Development as an Innovation Process



*Note the “red slices” for every level where the need for individual creativity is particularly critical*

## ... but Wouldn't Phase 4 Volatility Gradually Decrease?

- One might intuitively think that when a CTE matures, the uncertainty associated with the needed innovation gradually decreases and estimation predictability increases
- However
  - *This idea is based on the idealistic assumption that all CTEs are totally independent. In reality, mainly at and above TRL 6, we have to worry about potential interference from other CTEs or in some cases even from other, non-critical system elements*
    - This potential for interference is gradually increasing as the expanding structural context dimension of the rating scheme shows
- The need for creative, innovative thinking is not decreasing
  - *In fact, CTEs often have to be recursively re-evaluated or even fundamentally changed as their maturity evaluation and the overall system implementation progress*

## And Finally, an Expert's Voice ...

- Richard Feynman\* was once asked how many process steps he used during problem solving. After some thinking, he said that his “secret” process formula constituted the following three steps:
  - 1) *Write down the problem*
  - 2) *Think real hard*
  - 3) *Write down the solution*



*\* Richard Feynman (1918-1988); professor and researcher; one of the most famed physicist of the world. Besides his work on the atomic bomb at the Manhattan Project, he became also known as a key member of the panel that investigated the Space Shuttle Challenger disaster. The included picture is a copy of his old government ID from Los Alamos (Photograph reprinted courtesy of the United States Government.)*

# TRA and Risk Management Revisited

- We made an effort to neatly separate TRA and Risk Management
  - *However, there are various situations for interplay, synergy, and even conflict*
- Example for synergistic relationship
  - *There are numerous items that are not intentionally in-scope for a TRA, but the IRT might get insight into during attendance of the contractors' design reviews, studying research documents, etc.*
  - *It is the IRT members' duty to go beyond the core reporting on critical technologies and provide risk warnings to program management on any technology-related risk that has been discovered during a TRA*
- Example for interplay
  - *In case of COTS, TRA is viewed as an early screening activity to narrow down the list of COTS options for trade studies and risk reduction activities*
- Example for conflict
  - *Selection of a programming language/compiler for flight software (details to follow.)*

# Ada or C++ For Developing High Assurance Software?

## Level of Protection Within the Languages\*

| Criteria               | C++  | Ada 95 |
|------------------------|------|--------|
| Wild Jumps             | Some | Few    |
| Overwrites             | Some | Few    |
| Semantics              | Some | Good   |
| Model of Maths         | None | Good   |
| Operational Arithmetic | None | Good   |
| Data Typing            | Some | Good   |
| Exception Handling     | Some | Good   |
| Safe Subsets           | None | Good   |
| Exhaustion of Memory   | Some | Rare   |
| Separate Compilation   | Some | Good   |

*\* The issue is that there is a higher probability of violating good programming practices with C++ than with a more declarative language like Ada 95. For criteria details please see next slide.*

# Criteria for Ada/C++ Comparison

- Wild Jumps
  - *The program cannot jump to an arbitrary memory location*
- Overwrites
  - *The program cannot overwrite an arbitrary memory location*
- Semantics
  - *The compiler supports static code analysis*
- “Model of Maths”
  - *A rigorous model for floating-point and integer arithmetic*
- Operational Arithmetic
  - *Run-time checking to ensure adherence to “model of maths”*
- Data Typing
  - *Typing model is strong enough for static checking and run-time checking*
- Exception Handling
  - *Mechanisms to facilitate recovery from run-time faults are provided*
- Safe Subsets
  - *A language subset exists that increases the safety of the language*
- Exhaustion of Memory
  - *A mechanism exists to ensure that the program does not run out of memory*
- Separate compilation
  - *Type checking is provided across all parts of the development environment*



# Analysis: Technology vs. Programmatic Risks with Ada

- Even though it seems that Ada is superior for high-assurance, embedded system development, there are several reasons for the program manager to vote against Ada (Factors not supposed to be considered during a TRA):
  - *Ada programming skills are becoming scarce*
    - Universities shun Ada; they believe that there is no market for Ada skills
    - There is a disdain toward Ada amongst programmers that is partially a backlash due to the earlier DOD Ada mandate
    - Ada programmers have difficulty in getting jobs outside of DoD
  - *The available selection of development environments and tools for Ada is becoming narrower and more expensive*
- We need to balance between what the technology buys vs. other factors that could be more relevant in assessing programmatic risks
  - *The programmatic risks associated with adopting Ada may be higher than with adopting C++ even though Ada clearly poses a lower technology risk*

# Software Technology Maturity is not Software Maturity

- Software Technology Maturity definition revisited
  - *A measure of degree to which proposed concepts, processes, methods, algorithms, or tools, whose primary purpose is the development, operation, understanding, and maintenance of software, meet program objectives*
- Software Maturity (Software Product Maturity or Technical Maturity) characteristics are drastically different
  - *The form of software evolves rather than being replaced by artifacts of different forms (note the breadboards → brassboards → manufactured items metamorphosis in case of hardware)*
    - In most cases – unlike hardware – software is developed iteratively
      - *All software artifacts continuously and concurrently evolve*
  - *Software integration is different from hardware integration*
    - Software integration is a continuous activity from the beginning of the evolution of the objective system
      - *One does not “make” every component first and then “integrate”*
  - *Software Product Maturity is not an ever-increasing measure*
    - Software can become “senile”
      - *At some point in its life cycle, the number of changes and bug-fixes can overwhelm the architecture and the software starts degrading*

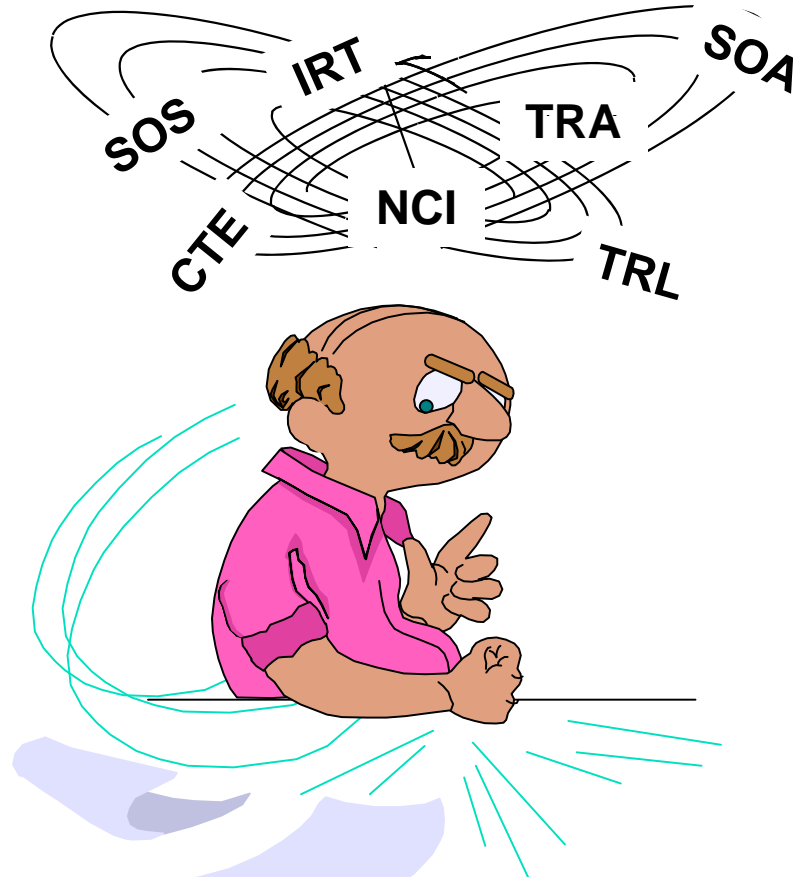
# Software Technology Maturity is not Software Maturity (cont.)

- The distinction is not simply about mincing words
  - *Technology maturity has an important gating role leading up to MS B, while planning and monitoring software product evolution are essential after MS B.*
- However, the current TRL definitions conflate these topics
  - *TRLs 1-4 focus on traditional technology innovation-type activities*
  - *TRLs 5-7 focus on the fidelity of test/validation environments in relation to integration and Verification and Validation (V&V) type activities*
  - *TRLs 8-9 focus on productization and deployment activities*
- Is this a problem for hardware elements of a program?
  - *Maybe, but this tutorial is about software ☺*
- Is this a problem for software elements of a program?
  - Yes. TRLs 1-4 reflect indeed software technologies, but the shift to V&V between TRL 4 and 5 is meant to be applied to a software product rather than a software technology

# Concluding Thoughts

- Technology Readiness Assessment, a critical tool in Defense Acquisition, helps us to prevent the incorporation of immature technologies during system design
- Modern weapon systems are software-intensive systems; consequently, the early evaluation of software technologies is essential
- Software Technology Readiness Assessments are carried out by software specialists. However
  - *An understanding of the maturity of all related, hardware technologies is necessary*
  - *During Technology Readiness Assessments a careful coordination amongst the technical area specialists is also a must*

# I am Sorry if You are Still Confused ...



I am sure I have failed to anticipate or answer all of your questions. I am sorry if it seems that the answers I provided only served to raise a new set of questions. In some ways you might feel that you (and I...) are still as confused as ever, but I do believe that now we are confused on a much higher level and about more important things...Thank you again for attending the tutorial!

# Acronyms

|       |  |
|-------|--|
| ACQ   | Acquisition  |
| API   | Application Programming Interface                  |
| APO   | Acquisition Program Office                         |
| ASIC  | Application Specific Integration Circuit           |
| AT&L  | Acquisition, Technology, & Logistics               |
| CASE  | Computer-Assisted Software Engineering             |
| CM    | Configuration Management                           |
| CMMI  | Capability Maturity Model Integration              |
| COTS  | Commercial Off-the-Shelf                           |
| CTE   | Critical Technology Element                        |
| DAPA  | Defense Acquisition Performance Assessment         |
| DBMS  | Data Base Management System                        |
| DDR&E | Director of Defense Research & Engineering         |
| DE    | Development Environment                            |
| DOD   | Department of Defense                              |
| DAB   | Defense Acquisition Board                          |
| FPGA  | Field-Programmable Gate Array                      |
| GOTS  | Government Off-the-Shelf                           |
| GUI   | Graphical User Interface                           |
| HW    | Hardware   |
| IEC   | International Electrotechnical Commission          |
| IRT   | Independent Review Team                            |
| ISO   | International Standards Organization               |
| JWS   | Joint War-fighting Space                           |
| KDP   | Key Decision Point                                 |
| MDD   | Model Driven Development                           |
| MIL   | Military   |
| MOIE  | Mission-Oriented Investigation and Experimentation |

|        |  |
|--------|--|
| NCI    | Network Centric Infrastructure             |
| NCO    | Network Centric Operations                 |
| NCW    | Network Centric Warfare                    |
| NSS    | National Security Space                    |
| NSSAP  | National Security Space Acquisition Policy |
| OFT    | Office of Force Transformation             |
| OMG    | Object Management Group                    |
| ORS    | Operational Responsive Space               |
| OS     | Operating System                           |
| OT&E   | Operational Test & Evaluation              |
| OUSD   | Office of the Under Secretary of Defense   |
| PL     | Programming Language                       |
| PSP    | Personal Software Process                  |
| R&D    | Research & Development                     |
| RHBD   | Radiation Hardening by Design              |
| RHBP   | Radiation Hardening by Process             |
| RPC    | Remote Procedure Call                      |
| SAF/AQ | Secretary of the Air Force/Acquisition     |
| SEAM   | Systems Engineering Assessment Model       |
| SMC    | Space and Missile Systems Center           |
| SOA    | Service Oriented Architecture              |
| STD    | Standard                                   |
| SW     | Software                                   |
| TRA    | Technology Readiness Assessment            |
| TRL    | Technology Readiness Level                 |
| UML    | Unified Modeling Language                  |
| USC    | United States Code                         |
| V&V    | Verification & Validation                  |

# References

- Barnaby 2004** Barnaby, H. J., *Will Radiation-Hardening-by-Design (RHBD) Work?*, IEEE Nuclear & Plasma Sciences Society News, September 2004
- Basadur 2001** Basadur, M., *The Power of Innovation: How to Make Innovation a Way of Life and How to Put Creative Solutions to Work*, Applied Creativity Press, Toronto Canada 2001
- Booch 1999** Booch, G., et al, *The Unified Modeling Language User Guide*, Addison-Wesley 1999
- DAPA 2006** Defense Acquisition Performance Assessment Report, March 2006
- DOD 2009** DOD Technology Readiness Assessment (TRA) Deskbook, July 2009
- Foreman 1997** Foreman, J., et al, *Software Technology Review*, CMU/SEI draft, June 1997
- ISO/IEC 2007** Recommended Practice for Architectural Description of Software-Intensive Systems, ISO/IEC 42010:2007
- Minkiewicz 2007** Minkiewicz, A., *The Cost and Business Impact of SOA – Is it Right for You*, A Price Systems Thought Leadership Article on the Price Systems website
- OUSD 2008** OUSD(AT&L) Memorandum for Service Acquisition Executives on the Implementation of Service Oriented Architecture (SOA) within DOD Acquisition Community, 25 July 2008
- Pew 2007** Pew, R., & Mavor, A., *Human-System Integration in the System Development Process: A New Look*, National Academies Press, 2007
- Rocke 2006** Rocke, P., et al, *Net Centric Enterprise Services (NCES) Applicability to Service Oriented Architectures within the DOD Satellite Command Community of Interest (COI)*, Ground System Architectures Workshop (GSAW) 2006
- Schuler 2005** Schuler, M. A., *Joint Warfighting Space and C2 of Deployable Space Forces*, High Frontier – The Journal for Space & Missile Professionals, Vol. 1, No. 4, 2005



# Backup Slides



# Relevant Environment for Space

- **Space Environment Attributes**
  - *Pressure (vacuum)*
  - *Temperature (deep space)*
  - *Gravity (microgravity on orbit)*
  - *Radiation*
  - *Electromagnetic Spectrum (star light, sunlight, x-ray, etc.)*
- **Launch Environment Attributes**
  - *Pressure (e.g., changing pressure, venting)*
  - *Temperature (e.g., payload fairing heating)*
  - *Gravity (acceleration,) Vibration, Structural Loads*
- **Designed Environment Attributes**
  - *Physical internal environment (e.g., self generated heating)*
  - *Software interaction with environment (processor throughput, memory capacity, bus bandwidth, etc.)*
  - *Hardware Interfaces (e.g., thermal shorts)*
  - *Behavior of model or prototype in interfacing environment (dynamic behavior)*
- **Operational Environment Attributes**
  - *Lifetime, Duty Cycles, Complex Dynamic Behavior*

# TRL 1-3 Evaluation Dimensions

| TRL | Basic TRL DEFINITIONS   | Artifacts   | Structural Context | SW Environment           | Validation Environment and Methods                        | Data Used for Validation      | Configuration Management | Documentation   |
|-----|---|---|--------------------|--------------------------|---|-------------------------------|--------------------------|---|
| 1   | Basic principles observed and reported  | Research articles, peer-reviewed white papers, point papers, early conceptual models                                | n/a                | "Academic", experimental | Basic research using analytical Methods                   | n/a                           | n/a                      | Appropriate and sufficient to demonstrate basic principles  |
| 2   | Technology concept and/or application formulated                                      | Analytic studies, papers on competing technologies  | n/a                | "Academic", experimental | Applied research using analytical Methods                 | Synthetic data only           | n/a                      | Appropriate and sufficient to demonstrate application concept   |
| 3   | Analytical and experimental critical function and/or characteristic proof of concept. | Analytical and simulation models; Availability of appropriate COTS/GOTS or reusable software artifacts is explored. | n/a                | "Academic", experimental | Active R&D initiated via the use of models and simulation | Partially representative data | n/a                      | Appropriate and sufficient to interpret analytical or experimental data; Full documentation is available on COTS/GOTS or reusable software under consideration. |

These tables help to evaluate all factors associated with a TRL

# TRL 4-6 Evaluation Dimensions

| TRL | Basic TRL DEFINITIONS   | Artifacts   | Structural Context     | SW Environment           | Validation Environment and Methods   | Data Used for Validation                                  | Configuration Management                                 | Documentation   |
|-----|---|---|------------------------|--------------------------|--|---|--|---|
| 4   | Component and/or breadboard validation in a laboratory environment          | A stand-alone prototype solving a partial-scale problem; Proposals for a nominal software architecture and for a simulation/stimulation work-up plan; COTS/GOTS, or reusable SW if applicable | Prototype SW Component | "Academic", experimental | Advanced technology development with throwaway or evolutionary SW prototypes | Representative data                                       | Limited scope; appropriate for experimental environment  | Appropriate and sufficient to interpret experimental results; Full documentation on chosen COTS/GOTS or reusable software |
| 5   | Component and/or breadboard validation in a relevant environment            | A stand-alone prototype solving a partial-scale problem; Detailed software architecture and final simulation/stimulation work-up plan; COTS/GOTS, or reusable SW if applicable                | Prototype SW Component | Operational-like         | Advanced technology development with throwaway or evolutionary SW prototypes | Representative data                                       | Appropriate for operational-like, production environment | Appropriate and sufficient to interpret results; Full documentation on chosen COTS/GOTS or reusable software              |
| 6   | System/subsystem model or prototype demonstration in a relevant environment | Viable prototype providing the foundation for productization  | Prototype WBS Level 3  | Operational-like         | Development using evolutionary SW Prototype                                  | High-fidelity data representative of relevant environment | Appropriate for operational-like, production environment | Appropriate and sufficient to validate relevant environment and interpret demonstration results                           |

*The red border emphasizes the special position TRL 6 occupies in the Technology Readiness Assessment Process*

# TRL 7-8 Evaluation Dimensions

| TRL | Basic TRL DEFINITIONS  | Artifacts             | Structural Context | SW Environment        | Validation Environment and Methods                         | Data Used for Validation                                  | Configuration Management               | Documentation   |
|-----|--|-----------------------|--------------------|-----------------------|--|---|--|---|
| 7   | System prototype demonstration in an operational environment         | Productized component | WBS Level 2        | Actual SW Environment | End-to-end testing of Production SW using system simulator | High-fidelity data representative of relevant environment | Appropriate for the actual environment | Appropriate and sufficient to validate test results   |
| 8   | Actual system completed and qualified through test and demonstration | Productized component | WBS Level 1        | Actual SW Environment | Testing of Production SW using OT&E                        | Real data   | Appropriate for the actual environment | Appropriate and sufficient to validate technology's performance and operate and maintain the product. |

# TRL 9 Evaluation Dimensions

| TRL | Basic TRL DEFINITIONS                                      | Artifacts             | Structural Context | SW Environment        | Validation Environment and Methods | Data Used for Validation | Configuration Management               | Documentation   |
|-----|--|-----------------------|--------------------|-----------------------|------------------------------------|--------------------------|--|---|
| 9   | Actual system proven through successful mission operations | Productized component | WBS Level 1        | Actual SW Environment | Actual Mission                     | Real data                | Consistent with the actual environment | Appropriate and sufficient to validate technology's performance and operate and maintain the product. |

# Use of Trademarks, Service Marks and Trade Names

Use of any trademarks in this material is not intended in any way to infringe on the rights of the trademark holder. All trademarks, service marks, and trade names are the property of their respective owners.